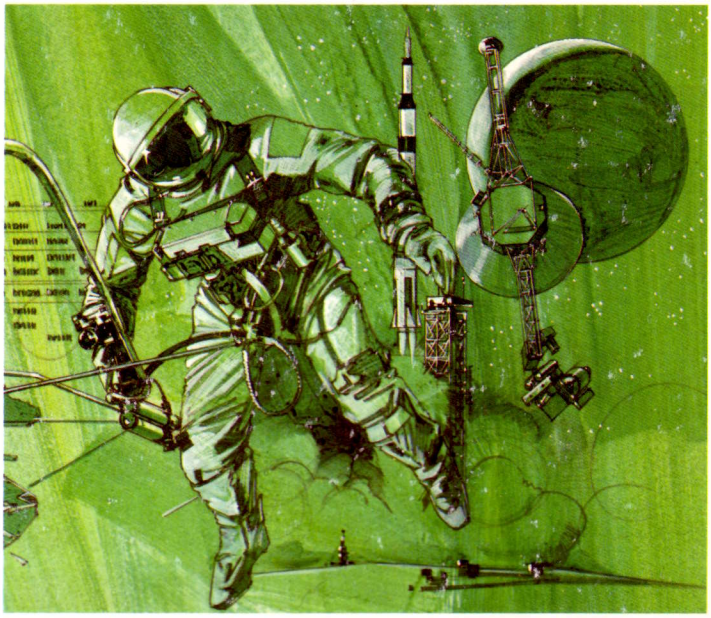
Personal Computer

MZ-800

BEDIENERHANDBUCH



SHARP®

Personal Computer

MZ-800

Bediener-handbuch

© SHARP CORPORATION

HINWEIS

Das vorliegende Handbuch beschreibt das Kassetten-BASIC 1Z-016 in Verbindung mit dem SHARP Personal-Computer MZ-800.

(1) Alle Systemsoftware für den MZ-800 wird in Form von Dateien auf Kassetten oder Disketten zur Verfügung gestellt. Der Inhalt dieser Dateien und dieses Handbuches kann ohne Vorankündigung geändert werden. Daher sollten die Versionsnummern der Programme jeweils sorgfältig geprüft werden.

(2) Die gesamte Systemsoftware des MZ-800 wurde von der Firma SHARP entwickelt. Sie ist daher ihr Eigentum und unterliegt dem Urheberrecht. Die Anfertigung von Kopien der Systemsoftware oder dieses Handbuches ist nicht gestattet.

(3) Der Computer und dieses Handbuch wurden vor der Versendung mit großer Sorgfalt geprüft. Sollten trotzdem Probleme oder Fragen auftauchen, deren Lösung sich nicht aus dem Handbuch ergibt, dann zögern Sie bitte nicht, sich mit Ihrem zuständigen SHARP-Händler in Verbindung zu setzen. Er wird Ihnen gern bei der Klärung von Fragen behilflich sein.
Dennoch kann SHARP für Schäden, die sich aus dem Gebrauch des MZ-800 ergeben, nicht haftbar gemacht werden.

Vorwort

Herzlichen Glückwunsch zu Ihrem Kauf des MZ-800. Ihr MZ-800 ist ein kompakter Personal Computer, zu dessen Ausstattung volle Grafikfähigkeit (Auflösung: 640 × 200 Punkte), Anzeige von 16 Farben und 3-Ton-Akkorde über 6 Oktaven gehört. Zu den herausragenden Merkmalen des MZ-800 zählt ferner, daß spezielle Hardware-Einrichtungen die Verwendung des MZ-800 anstelle eines Computers der Serie MZ-700 ermöglichen. Mit anderen Worten sind die meisten der für die SHARP Computer der Serie MZ-700 geschriebenen Programme auf dem MZ-800 ablauffähig. Dieses Handbuch ist so abgefaßt, daß es gleichzeitig als Anleitungsbuch für den MZ-800 und als BASIC Referenzhandbuch verwendet werden kann. Es wird Ihnen helfen, sich mit dem Computer und der dafür vorgesehenen Programmiersprache BASIC vertraut zu machen. Das Handbuch ist wie folgt aufgebaut:

Kapitel 1 enthält Hinweise zum Auspacken und zur Handhabung des Gerätes, Aufstellanleitungen und Hinweise dazu, welche Maßnahmen beim eventuellen Auftreten von Schwierigkeiten ergriffen werden sollen.

Im Kapitel 2 ist die Verfahrensweise zum Einschalten des Gerätes, Laden des BASIC Interpreters und Ausschalten des Gerätes beschrieben.

Der BASIC Interpreter wird im Kapitel 3 beschrieben.
In diesem Kapitel behandeln wir auch das Schreiben eines einfachen Programms einschließlich der Vornahme von Änderungen, Speichern auf Cassette und das Laden dieses Programms in den Arbeitsspeicher.

Im 4. Kapitel werden die Funktionen der Tasten auf der Tastatur beschrieben. Ferner werden in diesem Kapitel der Betrieb des Datenrecorders und die Handhabung der Cassette beschrieben.

Im Kapitel 5 befassen wir uns mit den zum Schreiben von Programmen erforderlichen Grundkenntnissen.

In Kapitel 6 werden BASIC-Befehle und Anweisungen behandelt.

Kapitel 7 zeigt die Hardware-Konfiguration des MZ-800 und die I/O-Port-Steuerung (I/O-Port = Eingabe/Ausgabe-Anschluß). Desgleichen werden die Peripheriegeräte beschrieben und wie man sie anschließt.

In Kapitel 8 wird das Monitor-Programm erklärt, das direkten Zugriff auf den Speicher möglich macht.

In Kapitel 9 befassen wir uns mit der MZ-700 Betriebsart des MZ-800.

Bitte die Hinweise zur Handhabung und die Aufstell-Anleitung vor Betätigung des Netzschalters am Computer auf jeden Fall durchlesen.
Damit Sie mit Ihrem MZ-800 die optimale Leistung erzielen, empfehlen wir ein möglichst gründliches Studium dieses Handbuchs.

INHALT

Kapitel 1 Einleitung
1.1 Hinweise zum Auspacken 1-2
1.2 Hinweise zur Handhabung 1-3
1.3 Außenansicht 1-4
1.4 Aufstellung 1-6
1.5 Auftreten von Problemen 1-10

Kapitel 2 Betriebsstart
2.1 Einschalten des Gerätes 2-2
2.2 Ausschalten des Gerätes 2-3
2.3 Ausführung des Demonstrationsprogramms 2-4

Kapitel 3 Grundlagen zur Bedienung
3.1 Einleitung 3-2
3.2 Die Tastatur 3-4
3.3 Schreiben eines einfachen Programms 3-6
3.4 Editieren von Programmen 3-9
3.5 Speichern eines Programms 3-12
3.6 Laden eines Programms 3-14

Kapitel 4 Tastatur und Cassettenlaufwerk
4.1 Die Tastatur 4-2
4.2 Cassettenlaufwerk 4-6

Kapitel 5 Programmier-Konzepte
5.1 Mehrere Anweisungen enthaltende Zeilen und Zeilennummern 5-2
5.2 Numerische Daten und Zeichenfolge-Daten 5-3
5.3 Konstanten 5-4
5.4 Variablen 5-5
5.5 Feldvariablen 5-7
5.6 Ausdrücke 5-8
5.7 Dateien 5-10
5.8 Funktionen 5-11
5.9 Bildschirm koordinaten 5-16

Kapitel 6 MZ-800 BASIC Befehle und Anweisungen
6.1 Befehle 6-3
6.2 Grundlegende Anweisungen 6-10
6.3 Dateisteuerbefehle 6-48
6.4 Grafik-Steuerbefehle 6-65
6.5 Musiksteuer-Befehle 6-78
6.6 Druckersteuer-Befehle 6-84
6.7 Maschinensprache-Steuerbefehle 6-96
6.8 Fehlerverarbeitungs-Befehle 6-100

Kapitel 7 Hardware

7.1 Hardware des MZ-800 7-2
7.1.1 System-Konfiguration 7-2
7.1.2 Systemschalter-Einstellungen 7-3
7.1.3 I/O Port-Steuerung 7-4
7.2 Peripheriegeräte 7-8
7.2.1 Standard-Schnittstellen 7-8
7.2.2 I/O-Erweiterungs-Anschlußbuchse 7-8
7.2.3 RAM-Datei-Steckkarte (MZ-1R18) 7-11
7.2.4 Joystick 7-12
7.2.5 Drucker 7-14
7.2.6 Optionaler Grafikspeicher MZ-1R25 7-18
7.2.7 Externer Cassettenrecorder (nur für MZ-811) 7-20

Kepitel 8 Monitor

8.1 Allgemeines 8-2
8.2 ROM-Monitor und BASIC-Monitor 8-3
8.3 Starten des ROM-Monitors 8-4
8.4 Monitorbefehle 8-5
8.5 BASIC-Monitor 8-8
8.6 BASIC-Monitor-Befehle 8-9

Kapitel 9 Betriebsart MZ-700

9.1 Verwendung von Programmen für den MZ-700 9-2
9.2 Zusammenfassung der MZ-700 BASIC-Befehle und Anweisungen, Funktionen und Operationen 9-3

Anhang

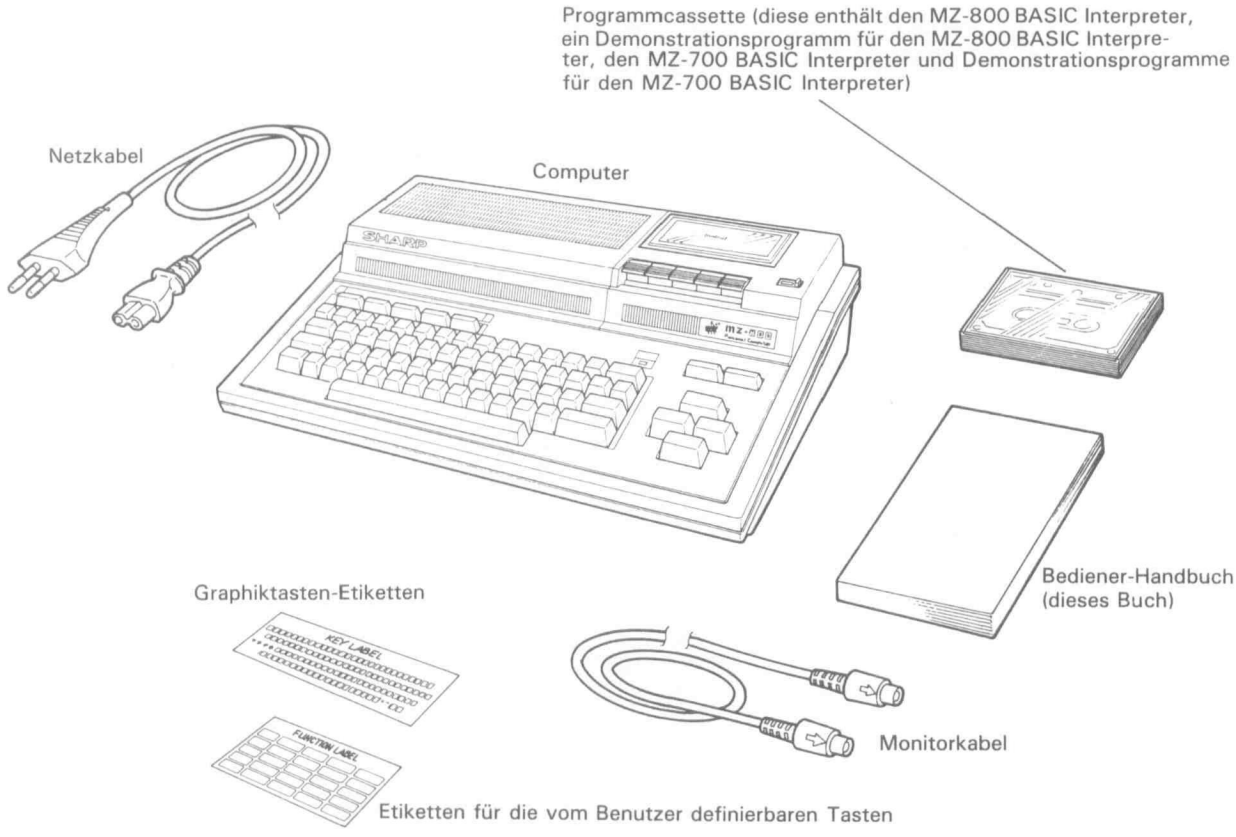
Anhang A Anzeigensteuerung bei MZ-800 Betrieb A-2
B Programmierbarer Tongenerator A-7
C Reservierte Wörter A-10
D Konsolensteuercodes A-12
E Einschränkungen beim Gebrauch von Datei-I/O-Befehlen und Anweisungen A-13
F ROM-Monitor-Unterprogramme A-14
G Herstellung einer Sicherungskopie des BASIC-Interpreters A-18
H Farbplotter A-19
I Farbplotter-Steuercodes A-22
J Code Tabellen A-25
K Fehlermeldungen des Monitors A-27
L BASIC-Fehlermeldungen A-28
M Index A-30
N Technische Daten A-32

Kapitel 1 Einleitung

In diesem Kapitel befassen wir uns mit der Handhabung und der Aufstellung des Computersystems. Vor dem ersten Einschalten des Gerätes sollte man dieses Handbuch auf jeden Fall sorgfältig lesen.

1.1 Hinweise zum Auspacken

Nach dem Auspacken sicherstellen, daß die folgenden Gegenstände enthalten sind:



Für eventuell erforderlichen späteren Transport des Computers empfiehlt es sich, das Original-Verpackungsmaterial aufzubewahren.

1.2 Hinweise zur Handhabung

1) Der Computer enthält zahlreiche Präzisionsteile. Das Gerät daher nicht an extremen heißen oder kalten Orten aufbewahren, bzw. an Orten mit raschen Temperaturwechseln. Desgleichen die Aufbewahrung des Computers an feuchten oder staubigen Orten vermeiden. Das Gerät nicht Chemikalien oder Gasen aussetzen.
2) Die Ventilationsöffungen nicht blockieren oder Gegenstände in deren Nähe plazieren.
3) Den Computer nicht Stößen oder Schwingungen aussetzen.
4) Direkte Sonneneinstrahlung auf den Computer vermeiden.
5) Kein Wasser oder sonstige Flüssigkeiten ins Gehäuseinnere eindringen lassen. Falls der Computer naß wird, ist es sehr gefährlich, ihn zu verwenden, außerdem besteht die Gefahr, daß elektronische Bauteile im Geräteinneren beschädigt werden.
6) Das Gehäuse nicht selbst öffnen bzw. zerlegen, es sei denn, man nimmt den von SHARP gegebenen Anleitungen entsprechend die Installation von Optionen vor.
7) Wenn ein Radio- oder Fernsehgerät in der Nähe des Computers verwendet wird, so wird der Empfang von Sendungen möglicherweise elektronisch gestört. Daher solche Geräte (mit Ausnahme des als Computer-Bildschirmanzeige verwendeten) möglichst vom Computer fernhalten.
8) Beim Anschluß von Peripheriegeräten erscheint die Anzeige auf dem Bildschirm möglicherweise verzerrt. In diesem Falle empfiehlt es sich, die Anordnung der Systemkomponenten versuchsweise zu ändern.
9) Mit Ausnahme des als Option erhältlichen Plotters (MZ-1P16) keine Gegenstände auf dem Gehäuse plazieren.
10) Nach Ausschalten des Gerätes beim Ziehen des Netzsteckers stets am Stecker selbst, nicht am Kabel anfassen.
11) Wenn man den Computer nicht verwendet, sollte man auf jeden Fall den Netzschalter ausschalten. Nach Ausschalten des Gerätes mit dem Netzschalter mindestens 10 Sekunden warten, bevor man es wieder einschaltet, da andernfalls das System möglicherweise nicht einwandfrei arbeitet.
12) Zur Reinigung des Gerätes ein trocknes weiches Tuch verwenden, keine leicht flüchtigen Flüssigkeiten wie z.B. Alkohol oder Benzin benützen bzw. einen nassen Lappen. Die Nichtbeachtung dieser Vorsichtsmaßnahme kann zu Verfärbungen oder Verformungen des Gehäuses führen.
13) Wenn ungewöhnliche Bedingungen wie z.B. extrem hohe Temperaturen, ungewöhnlicher Geruch oder das Auftreten von Rauch zu bemerken ist, bitte umgehend den Betrieb beenden, das Gerät ausschalten und den Netzstecker ziehen.

MZ-811 und MZ-821

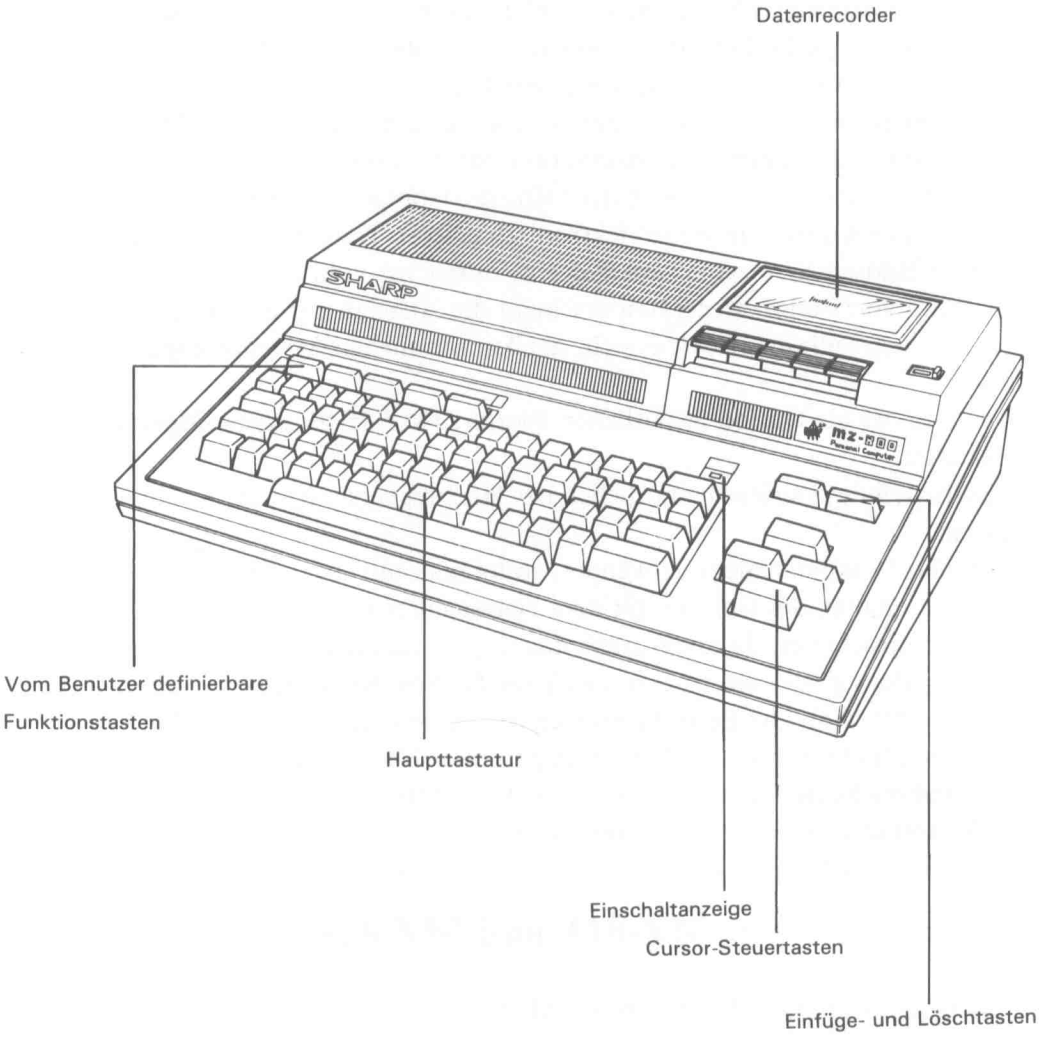
Eins der in diesem Handbuch beschriebenen Modelle ist möglicherweise in einigen Ländern nicht erhältlich.
In diesem Handbuch werden zwei Personal Computer Modelle erklärt:
Der MZ-811 und der MZ-821. Der Unterschied zwischen diesen beiden Computern ist wie folgt.

Modellname	MZ-811	MZ-821
Datenrecorder	Optional	Standard
Normaler Cassettenrecorder	Anschließbar	Nicht anschließbar

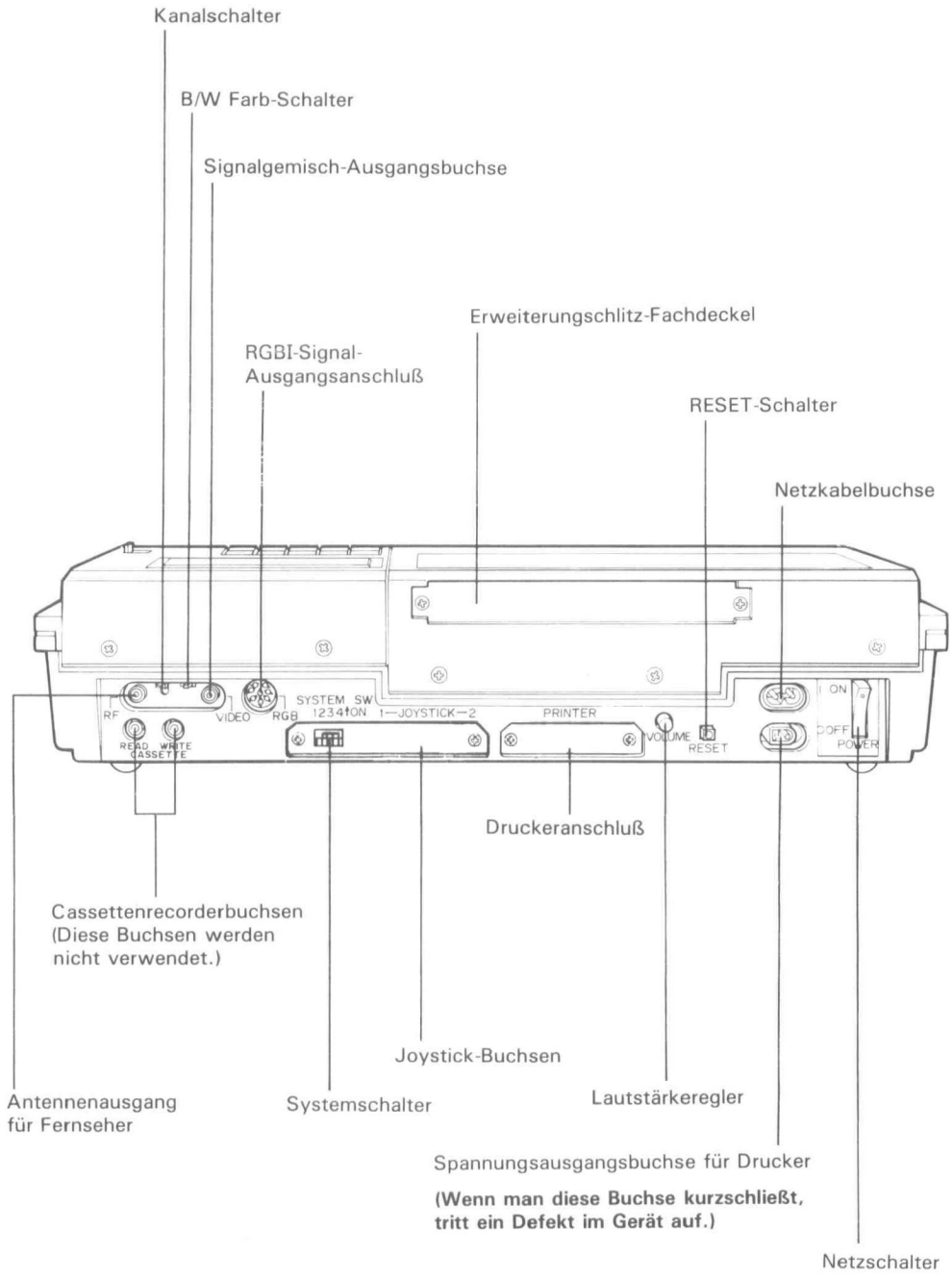
Durch Anschluß des optionalen Datenrecorders MZ-1T04 am MZ-811 wird er mit dem MZ-821 gleichwertig. Die Verfahrensweise zum Anschluß des Datenrecorders ist im Benutzerhandbuch für den MZ-1T04 beschrieben.
Die in diesem Handbuch gegebenen Erklärungen basieren auf dem MZ-821. Die Erklärungen auf den Seiten 7-3 und 7-20 gelten jedoch nur für den MZ-811.

1.3 Außenansicht

(Ansicht von vorn)

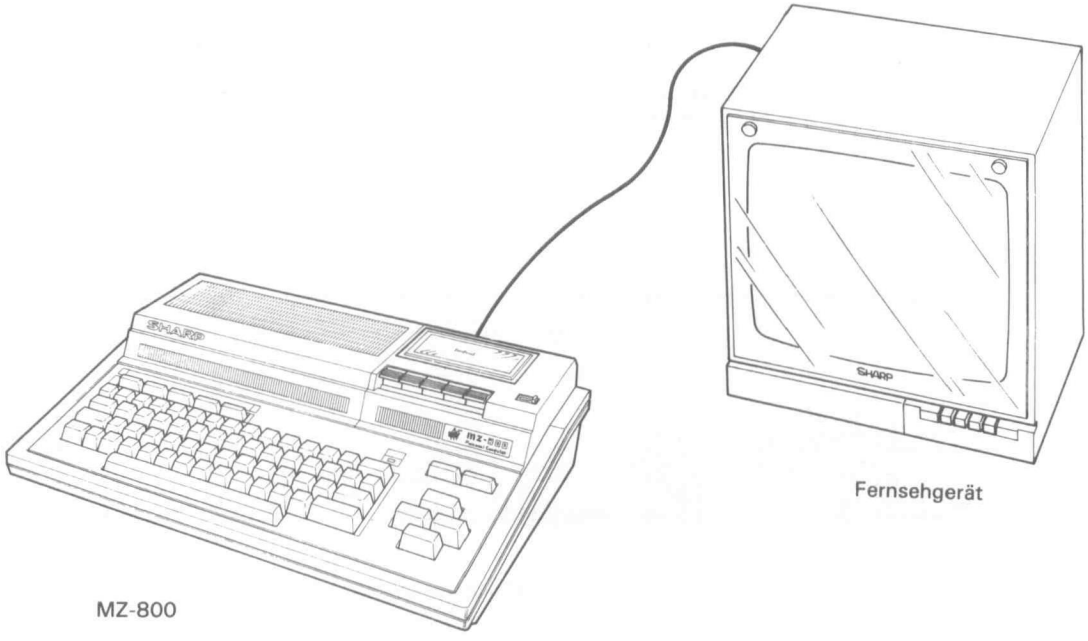


(Ansicht von hinten)



1.4 Aufstellung

Zum Betrieb des MZ-800 ist zunächst die Aufstellung des Systems erforderlich. Ferner wird eine Bildschirmanzeige benötigt, damit man sieht, was der Computer zur Zeit ausführt.
SHARP hat mehrere Ausführungen von Anzeigegeräten für den MZ-800 im Angebot. Man kann aber auch ein normales, für UHF-Empfang geeignetes Fernsehgerät als Bildschirmanzeige verwenden. Nachstehend ist die zum Betrieb des Computers erforderliche Mindestkonfiguration abgebildet.

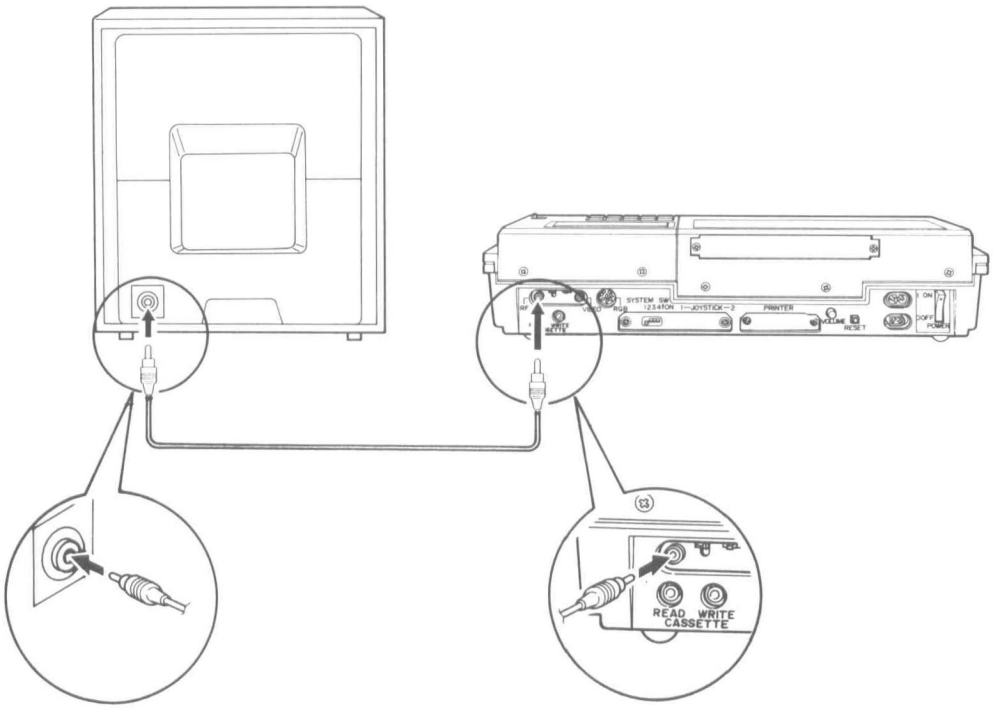


Vor der Betätigung des Netzschalters ist die Aufstellung des Systems erforderlich. Die nachstehenden Erklärungen beziehen sich auf das Verfahren beim Aufbau von typischen Systemen.

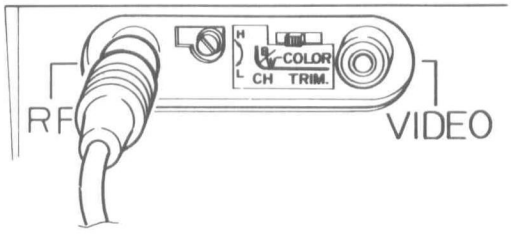
(1) Verwendung eines Fernsehgerätes

Wenn man vorhat, ein Fernsehgerät als Bildschirmanzeige einzusetzen, verwendet man das mit dem MZ-800 mitgelieferte Antennenkabel.

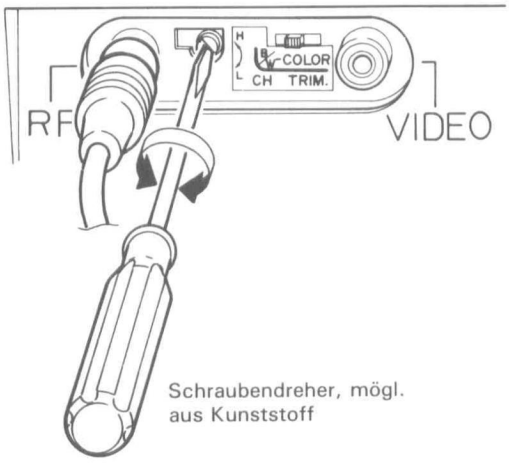
1) Alle Antennenkabel vom Fernsehgerät abtrennen, falls man sie angeschlossen läßt, sendet das Gerät möglicherweise unerwünschte Radiowellen, die sich störend auf den Empfang mit anderen Fernsehgeräten auswirken können.
2) Den PIN Stecker des Anschlußkabels schließt man an die RF PIN Buchse des MZ-800 an, das andere Ende des Kabels wird mit der 75-Ohm-UHF-Antennenbuchse des Fernsehgerätes verbunden.



3) Bei Verwendung eines Fernsehgerätes stellt man den Schalter für Schwarzweiß/Farb-Wahl "B/W-color" des MZ-800 auf COLOR, andernfalls läßt man den Schalter in der B/W-Stellung.



4) Den Kanalwähler des Fernsehgerätes stellt man auf einen Kanal zwischen 33 und 39 ein, der nicht durch einen Sender belegt ist. Erst das Fernsehgerät und dann den MZ-800 einschalten. Den Kanalregler so einstellen, daß die folgende Anzeige klar auf dem Bildschirm erscheint.



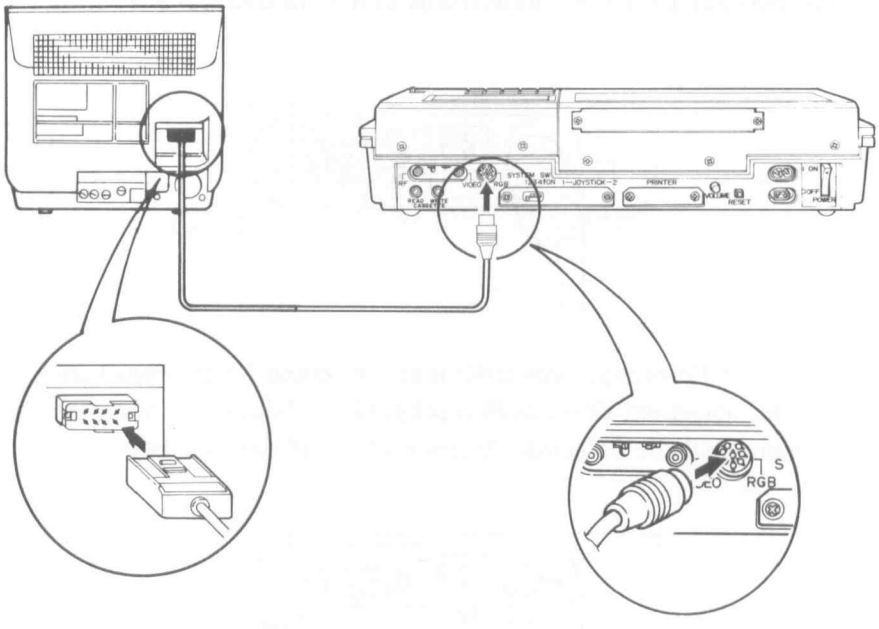
```
Make ready CMT
Please push key
C: Cassette tape
M: Monitor
```

Hinweis:

- Die mit einem normalem Fernsehgerät erzielte Bildqualität ist nicht so gut wie die mit den von Sharp erhältlichen Monitoren - bei manchen Fernsehgeräten erscheint ein Teil der Anzeige möglicherweise nicht auf dem Bildschirm. Dies liegt wahrscheinlich an der Einstellung des Fernsehgerätes. In diesem Falle wendet man sich an den zuständigen Fachhändler.
- Wenn die UHF-Antennenbuchse keine DIN-Buchse ist, verwendet man ein geeignetes Anschlußkabel mit einem pin-Stecker am einen Ende, und einem für das Fernsehgerät passenden Anschlußstecker auf der anderen Seite. Da die Impedanz des Hf-Ausgangssignals 75 Ohm beträgt, sollte man auf jeden Fall ein Kabel mit 75 Ohm Impedanz verwenden.
- Über den Hf-Signalausgang werden keine Tonsignale ausgegeben.

(2) Verwendung der Farb-Bildschirmanzeige MZ-1D19

1) Den rechteckigen Stecker des Kabels (gehört zur Bildschirmanzeige) an die Anschlußbuchse an der Rückseite der Bildschirmanzeige anschließen.
2) Den DIN-Stecker am anderen Ende des Kabels mit der RGB-Buchse an der Rückseite des MZ-800 verbinden.

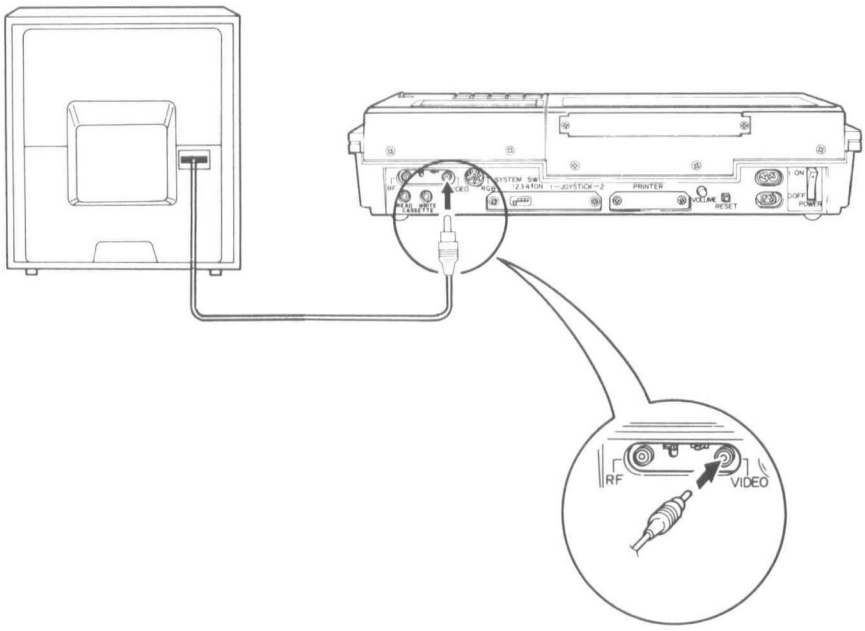


Hinweis:

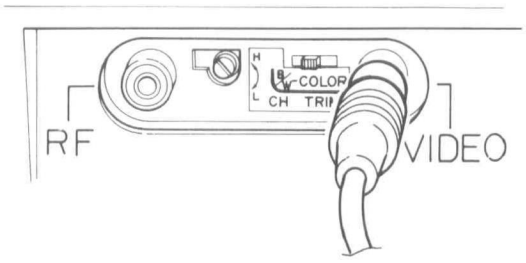
Auch ein mit RGB-Eingangsbuchse ausgestattetes Farbfernsehgerät kann man an die RGB-Buchse des MZ-800 anschliessen. In diesem Falle nimmt man zur Vorbereitung des Anschlußkabels Bezug auf die Bedienungsanleitung (RGB-Buchsenbelegung) des Fernsehgerätes.

(3) Verwendung einer grünen Bildschirmanzeige (MZ-1D04)

Den Stecker des Kabels für die grüne Bildschirmanzeige an die Videosignal-Ausgangsbuchse an der Rückseite des MZ-800 anschließen.



Den B/W-color Schalter auf B/W stellen.



Hinweis:
Ein mit Videoeingangsbuchse ausgestattetes Farbfernsehgerät kann man an die Videosignal-Ausgangsbuchse des MZ-800 anschließen. Zum Anschluß das mit dem MZ-800 mitgelieferte Kabel verwenden (Adapterstück von Chinch auf BNC-oder DIN-Buchse).

1.5 Auftreten von Problemen

Falls während des Betriebs mit dem MZ-800 Probleme auftreten, bitte zunächst diesen Abschnitt durchlesen, bevor man sich mit dem Händler in Verbindung setzt.
In der folgenden Tabelle sind mögliche Probleme und Prüfpunkte aufgeführt.

Problem	Prüfpunkt
Schlechte Bildqualität	• Ist das Anschlußkabel korrekt angeschlossen? • Ist der Kanalwähler des Fernsehgerätes so eingestellt, daß er mit der Einstellung des Kanal-Schalters vom MZ-800 übereinstimmt? (Siehe Seite 1-7) • Ist der B/W-color Schalter korrekt eingestellt?
Nichts erscheint auf dem Bildschirm.	• Ist der Netzschalters des Bildschirms eingeschaltet? • Ist das Netzkabel des Bildschirms an eine Steckdose angeschlossen? • Ist der Computer eingeschaltet?
Der Programmablauf kann nicht gestoppt werden.	• Zum Stoppen eines BASIC Programms die [BREAK] Taste drücken, wobei man die [SHIFT] Taste gedrückt hält. • Zum Stoppen eines Maschinensprache- oder Monitorprogrammms den RESET Schalter auf der Rückseite drücken.
Laden des Programms von der Cassette nicht möglich.	• Wurde das Laden korrekt ausgeführt? Das Ladeverfahren ist für Maschinensprache-Programme und BASIC-Programme jeweils unterschiedlich. Zum Laden eines Maschinensprache-Programms verwendet man das Monitor-Kommando L und zum Laden eines BASIC Programms den LOAD Befehl.
Sonstige Probleme	• Um den Betrieb nochmals von Anfang an auszuführen, den RESET Schalter an der Rückseite des Gerätes drücken.

Kapitel 2 Betriebsstart

2.1 Einschalten des Gerätes

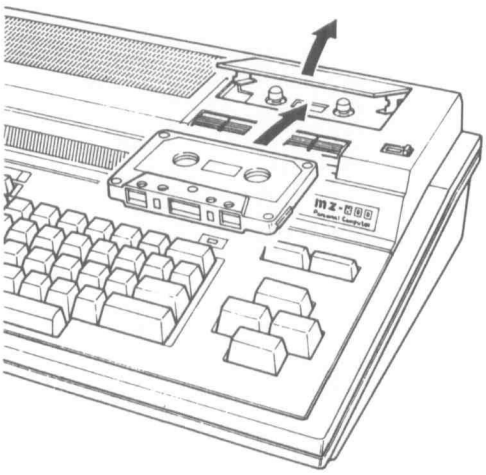
Zum Starten des Computersystems schaltet man zunächst den MZ-800 ein, dann die Bildschirmanzeige (falls weitere Peripheriegeräte angeschlossen sind, schaltet man auch diese nun ein).
Schalten Sie die Geräte in der folgenden Reihenfolge ein.

1. Den Computer MZ-800
2. Die Erweiterungseinheit (MZ-1U06)
3. Peripheriegeräte (z.B. den Drucker)

Es erscheint die folgende Anzeige auf dem Monitorbildschirm (nachstehend "Bildschirm" genannt):

```
Make ready CMT
Please push key
C: Cassette tape
M: Monitor
```

Bandlockerheit des Cassettenbands beseitigen (siehe Seite 4-7). Die Auswurftaste (EJECT) des MZ-800 Datenrekorders drücken. Dann die Cassette einschieben, wobei die mit BASIC 1Z-016 markierte Seite nach oben weisen sollte.



Den Deckel des Cassettenfachs herunterdrücken. Daraufhin die Taste C der Haupttastatur betätigen. (Durch Drücken der [M]-Taste wird der Monitor gestartet. Siehe Kapitel 8.) Die Bildschirmanzeige verändert sich wie folgt:

```
Make ready CMT
```

Die Wiedergabetaste (PLAY) des Cassettenrecorders drücken.
Die Bildschirmanzeige verändert sich wie folgt:

```
IPL is looking for a program
```

Daraufhin wird die folgende Meldung angezeigt:

```
IPL is loading MZ-1Z016
```

Es dauert nun ein paar Minuten, bis die folgende Anzeige auf dem Bildschirm erscheint:
Für Bandstop die Stoptaste [STOP] drücken.

```
BASIC interpreter 1Z-016 VX.XX
Copyright (C) 1984 by SHARP CORP.

XXXXX bytes free
Ready
■
└─Cursor (blinkt)
```

Der Computer ist jetzt zur Eingabe von Befehlen bereit. Der BASIC Interpreter ist nun in den Speicher eingelesen.

2.2 Ausschalten des Gerätes

Wenn die Meldung "Ready" angezeigt wird und der Cursor blinkt, kann man den Computer mit dem Netzschalter ausschalten. Während das Cassettenband läuft, sollte man jedoch den Netzschalter nicht betätigen.
Durch Ausschalten des Gerätes mit dem Netzschalter werden alle im Speicher stehenden Programme und Daten gelöscht. Daher sollte man auf jeden Fall vor Ausschalten des Gerätes alle benötigten Programme und Daten auf Cassette speichern. Die Verfahrensweise zum Speichern ist im Kapitel 3 beschrieben.
Schalten Sie die Geräte in der folgenden Reihenfolge aus.

1. Peripheriegeräte (z.B. Drucker)
2. Erweiterungseinheit (MZ-1U06)
3. Den Computer MZ-800

Hinweis:
Während der Datenrecorder in Betrieb ist, den MZ-800 nicht ausschalten.

2.3 Ausführung des Demonstrationsprogramms

Die mit dem MZ-800 mitgelieferte Cassette enthält ein Demonstrationsprogramm. Dieses Programm kann man wie folgt zur Ausführung bringen.
Nach Einlesen des BASIC Interpreters und nachdem das Band bis zum Zählwerkstand 170 durchgelaufen ist, gibt man folgendes ein:

RUN "CMT:" [CR]

Wenn auf dem Bildschirm die nachstehende Anzeige erscheint:

RUN "CMT:"
↓ PLAY

drückt man die Wiedergabetaste (PLAY)

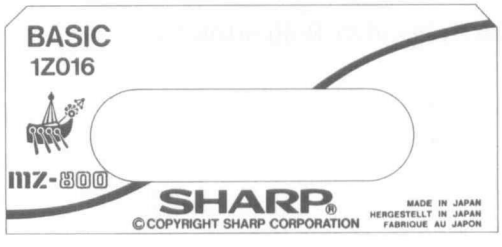
Das Demonstrationsprogramm wird nun geladen und ausgeführt.
Zum Stoppen des Programms drückt man die [SHIFT]- und die [BREAK]-Taste gleichzeitig. Auch nach Starten des Demonstrationsprogramms läuft das Band noch weiter. Nachdem der Bandlauf stoppt, die [STOP]-Taste drücken.

• Zubehör-Cassette

Die mit dem Computer mitgelieferte Zubehör-Cassette enthält die folgenden Dateien.

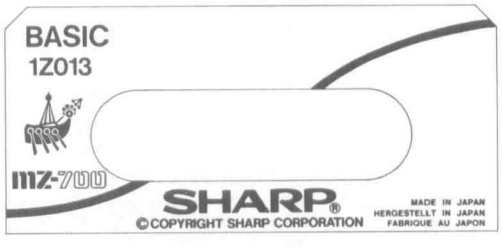
Seite A
"MZ-1Z016" ..... MZ-800 BASIC Interpreter (1Z-016)
"OPENING 800" ..... Demonstrationprogramm für MZ-800 BASIC
"OPENING DATA" ..... Daten für das Demonstrationsprogramm

Seite A Etikett



Seite B
"S-BASIC" ..... MZ-700 BASIC Interpreter (1Z-013)
"OPENING"
"MUSIC" .... Demonstrationsprogramme für MZ-700 BASIC
"COLOR PLOTTER"

Seite B Etikett



Kapitel 3 Grundlagen zur Bedienung

3.1 Einleitung

Ein Computer kann nur diejenigen Anweisungen ausführen, die der Benutzer zuvor in ihn eingegeben hat. Der MZ-800 ist ab Werk mit einem bestimmten Befehlssatz ausgestattet; nach Einschalten des Gerätes erfolgt die Betriebsweise jederzeit gemäß diesen vorgegebenen Anweisungen. Dieser Befehlssatz wird auch Monitorprogramm genannt und ist im ROM(*) gespeichert. Dieses bewirkt die Anzeige des ersten Bildschirms, woraufhin der Computer auf eine entsprechende Eingabe (Drücken einer der vorgeschriebenen Tasten) wartet. Je nachdem, welche Taste gedrückt wird, ermöglicht der Monitor die Ausführung von Monitorkommandos oder das Einlesen eines größeren Befehlssatzes von einem externen Speichergerät, z.B. von einem Cassettenrecorder, woraufhin dieser neue Befehlssatz in den RAM Speicher (*) eingelesen wird.

*: ROM und RAM sind Speichervorrichtungen zum Speichern der benötigten Daten. ROM steht für Nur-Lese-Speicher, der Inhalt dieses Speichers kann stets nur gelesen, nicht aber verändert oder gelöscht (z.B. durch Ausschalten des Gerätes) werden. RAM steht für Speicher mit wahlfreiem Zugriff, man kann ihn ganz nach Bedarf zum Lesen und Schreiben von Daten verwenden. Im MZ-800 befindet sich der Monitor in einem ROM-Speicher. In den RAM-Speicher werden BASIC Interpreter, BASIC Programm und Daten eingelesen. Der Monitor wird detailliert im Kapitel 8 erklärt, in diesem Kapitel befassen wir mit dem BASIC Interpreter. Alle Befehle müssen dem Computer in einer Sprache gegeben werden, die er versteht. Diese Sprache nennt man "Maschinensprache". Maschinensprache besteht aus einer Aneinanderreihung von Zahlen. Sie ist schwer zu verstehen, aber doch erlernbar.

Aber wir brauchen die Maschinensprache selbst auch gar nicht zu verstehen. Der BASIC Interpreter hilft uns weiter. BASIC ist eine am Englischen orientierte Programmiersprache, sie ist viel leichter verständlich als Maschinensprache. Der BASIC Interpreter übersetzt die in BASIC geschriebenen Befehle einzeln nacheinander in Maschinensprache, so daß der Benutzer mit Hilfe des BASIC Interpreters ablauffähige Programme in BASIC schreiben kann. Im Gegensatz zu Systemkommandos und sonstigen Anweisungen haben wir es bei BASIC hauptsächlich mit " Befehlen" zu tun. Wenn man während der ersten Bildschirmanzeige nach Einschalten des Gerätes die [C]-Taste drückt, lädt der Monitor den BASIC Interpreter von der Cassette in den RAM-Speicher, woraufhin der BASIC Interpreter aktiviert ist. "Laden" bedeutet, daß Daten von einer bestimmten Speichereinheit gelesen und in eine andere Speichereinheit eingegeben werden. Nach dem Laden des BASIC Interpreters erscheint zunächst die folgende Anzeige auf dem Bildschirm.

```
BASIC interpreter 1Z-016 VX.XX
Copyright (C) 1984 by SHARP CORP.

 XXXXXbytes free
Ready
■
(blinkt)
```

In diesem Zustand kann man den Computer im Dialogbetrieb verwenden. Man gibt einen Befehl ein, der Computer reagiert darauf, man gibt einen weiteren Befehl ein, der Computer reagiert usw. Wenn man einen falschen Befehl eingibt, antwortet der Computer darauf mit einer Fehlermeldung. Mit dieser Methode kann man jedoch den Computer nicht zur Erledigung von größeren, komplizierten Aufgaben veranlassen. Um den Computer zur Ausführung von komplexen Arbeiten einzusetzen, ist es erforderlich, ein Programm zu schreiben, das zur Ausführung im RAM-Speicher stehen muß. Ein Programm ist eine Aneinanderreihung von Befehlen, die vom BASIC Interpreter automatisch nacheinander interpretiert werden. Als BASIC Programm bezeichnen wir ein Programm, das unter Verwendung des BASIC Interpreters ausgeführt werden kann.

3.2 Die Tastatur

Nach Ausführung der vorbereitenden Maßnahmen zum Starten des Systems wie im Abschnitt 2.1 beschrieben, ist der Computer zur Ausführung der über die Tastatur eingegebenen Befehle bereit.

```
BASIC interpreter 1Z-016 VX.XX
Copyright (C) 1984 by SHARP CORP.

 XXXXXbytes free
Ready
■
(blinkt)
```

Auf dem Bildschirm ist die rechteckige, blinkende Schreibmarke zu sehen, der sog. Cursor. Drücken Sie nun eine beliebige Taste auf der Tastatur. Sie werden sehen, daß der Cursor nach rechts wandert und das entsprechende Zeichen an der Stelle auf dem Bildschirm erscheint, an der sich zuvor der Cursor befand.

Drückt man nun weitere Tasten, erscheinen die entsprechenden Zeichen jeweils auf dem Bildschirm. Der Cursor befindet sich stets rechts vom zuletzt eingegebenen Zeichen. Drücken Sie nun die mit "CR" markierte Taste, die sich im rechten Teil der Tastatur befindet. Diese für den Zeilenwechsel vorgesehene Taste bezeichnen wir als [CR]-Taste.
Was ist geschehen? Es erscheint die Meldung "Syntax error" auf der nächsten Zeile.

```
BASIC interpreter 1Z-016 VX.XX
Copyright (C) 1984 by SHARP CORP.

 XXXXXbytes free
Ready
ASDFGJJ
Syntax error
Ready
■
```

Die Meldung "Syntax error" weist darauf hin, daß der Computer den eingegebenen Ausdruck nicht verstehen kann. Der Grund dafür ist, daß der Computer nur eine ihm zu eigene Sprache verstehen kann, in diesem Falle BASIC. Zur Kommunikation mit dem Computer müssen wir daher BASIC verwenden.
Die Programmiersprache BASIC ist im Kapitel 6 erklärt, Fehlermeldungen enthält der Anhang G. Zu diesem Zeitpunkt brauchen wir noch keine speziellen BASIC Kenntnisse. Geben Sie als nächstes einmal den folgenden Satz unter Verwendung der Tastatur ein.

```
PRINT "ABC"
```

Die Leerstelle zwischen dem T in "PRINT" und dem ersten Anführungszeichen (") kann auch ausgelassen werden. Dann die [CR]-Taste drücken.
Auf dem Bildschirm erscheint die Zeichenfolge "ABC" unter dem eingegebenen Satz. Der Computer führt die Anzeige dieser Zeichen in Beantwortung des von Ihnen eingegebenen Befehls aus. Das Wort "PRINT" weist den Computer an, die von Anführungsstrichen eingeschlossene Zeichenfolge auf dem Bildschirm anzuzeigen. Wörter, die dem Computer befehlen, etwas Bestimmtes auszuführen, nennen wir Befehle oder Anweisungen.
Jede eingegebene Anweisung oder Befehl bewirkt, daß der Computer eine bestimmte Tätigkeit ausführt. Jedoch kann man mit diesem Verfahren nicht erreichen, daß der Computer komplexe Aktionen (z.B. das Spielen einer Melodie oder die Ausführung eines Computerspiels) vornimmt. Für solche Aufgaben ist ein Programm erforderlich.

3.3 Schreiben eines einfachen Programms

Zunächst wird durch Ausführung der im Abschnitt 2.1 beschriebenen Maßnahmen der BASIC Interpreter gestartet.

```
BASIC interpreter 1Z-016 VX.XX
Copyright (C) 1984 by SHARP CORP.

XXXXXbytes free
Ready
■
(blinkt)
```

Nun geben Sie einmal die folgenden Zeichen ein.

10 CLS

Die Leerstelle zwischen 0 und C kann auch ausgelassen werden. Dann die [CR]-Taste drücken. CLS ist eine Anweisung, die das Löschen des gesamten Bildschirms bewirkt, d.h. alle Zeichen werden vom Bildschirm gelöscht. Die Ausführung findet jedoch zunächst noch nicht statt, der Cursor blinkt, wie nachstehend gezeigt, zu Beginn der Zeile unterhalb von "10 CLS".

10 CLS
■

Die Leerstelle zwischen 10 und CLS kann auch ausgelassen werden. Die einer Anweisung vorangestellte Nummer ist die Zeilennummer. Wenn mehrere Anweisungen im Speicher abgelegt werden, dienen die Zeilennummern zur Festlegung der Reihenfolge, in der die Anweisungen interpretiert und an den Computer weitergegeben werden. Schreiben Sie jetzt einmal die folgenden Zeichen und drücken Sie die [CR]-Taste.

RUN

Daraufhin werden alle Zeichen vom Bildschirm gelöscht. RUN ist ein Befehl, der den BASIC Interpreter anweist, alle bis dahin im Speicher abgelegten Anweisungen der Reihenfolge der Zeilennummer entsprechend nacheinender in Maschinensprache-Befehle umzuwandeln und an den Computer weiterzugeben. Die CLS Anweisung befindet sich noch im Speicher. Dies kann man durch Eingabe des folgenden Befehls überprüfen.

LIST [CR]

[CR] bedeutet, daß die [CR]-Taste gedrückt werden soll. Der Bildschirm verändert sich wie folgt.

```
LIST
10 CLS
Ready
■
```

Nun geben wir einmal folgendes ein.

```
20 PRINT "BEISPIELPROGRAMM" [CR]
30 PRINT "MZ-800" [CR]
40 END [CR]
```

Wir wissen schon, wofür der PRINT Befehl dient. Die END Anweisung informiert den BASIC Interpreter über das Programmende. Durch Eingabe von RUN und Drücken der [CR]-Taste verändert sich der Bildschirm dann wie folgt.

```
BEISPIELPROGRAMM
MZ-800
Ready
■
```

Zur Anzeige des gesamten Programms geben wir nun das LIST Kommando ein.

```
BEISPIELPROGRAMM
MZ-800
Ready
LIST
10 CLS
20 PRINT "BEISPIELPROGRAMM"
30 PRINT "MZ-800"
40 END
Ready
■
```

Nun bitte folgendes Kommando eingeben:

NEW [CR]

Daraufhin löschen wir durch Eingabe des CLS Befehls den Bildschirm.

CLS [CR]

Jetzt geben wir nochmals LIST ein.

LIST [CR]

In diesem Falle wird kein Programm-Listing angezeigt. Mit NEW wurde nämlich das gesamte im Speicher stehende Programm gelöscht. Um daher ein neues Programm zu schreiben, sollte man stets zunächst NEW eingeben.
Geben Sie einmal folgendes Programm ein:

```
10 INPUT "A="; A
20 INPUT "B="; B
30 C=A+B
40 PRINT "A+B=";C
50 END
```

Dieses Programm berechnet die Summe von zwei Werten, die über die Tastatur eingegeben wurden. Mit diesem Programm stellen wir zwei neue Anweisungen vor, nämlich INPUT und LET (LET wird in abgekürzter Form durch das "=" Symbol in Zeile 30 ausgedrückt).
Der INPUT Befehl in Zeile 10 dient zum Einlesen eines über die Tastatur eingegebenen Wertes ins Programm als Variable A. Der INPUT Befehl in Zeile 20 liest den über die Tastatur eingebenen Wert ins Programm ein und bewahrt ihn als Variable B. Die LET Anweisung von Zeile 30 setzt die Summe von A und B mit der Variablen C gleich. Nach Schreiben des obigen Programms wird durch Eingabe des RUN Befehls die Bildschirmanzeige wie folgt verändert:

```
10 INPUT "A=";A
20 INPUT "B=";B
30 C=A+B
40 PRINT "A+B=";C
50 END
RUN [CR]
A=
```

Geben Sie nun eine beliebige Zahl ein und drücken Sie die [CR]-Taste. Daraufhin wird "B=" im Anschluß an die obige Zeile angezeigt.

```
A=35 [CR]
B=
```

Geben Sie eine weitere Zahl ein und drücken Sie die [CR]-Taste. Die Summe von A und B wird nun wie folgt angezeigt.

```
A=35
B=23 [CR]
A+B= 58
```

Die bisher gezeigten Programme sind extrem einfach aufgebaut; schreiben Sie nun versuchsweise eigene Programme , indem Sie die im Kapitel 4 aufgeführten Befehle und Anweisungen nach Wunsch kombinieren.
Hinweis zur Begriffsbestimmung: grundsätzlich werden die Ausdrücke "Befehle" und "Anweisungen" gleichbedeutend behandelt. Eine Sonderstellung nimmt der Begriff "Kommando" ein. Ein Kommando (auch "Direktbefehl" genannt) wird ohne Zeilennummer eingegeben und unmittelbar ausgeführt.

3.4 Editieren von Programmen

Beim Programmieren auftretende Schreibfehler kann man leicht wieder korrigieren. Der BASIC Interpreter ermöglicht das Editieren eines im Speicher stehenden Programms.
Jeder beliebige Teil eines Programms kann editiert werden, man muß ihn hierfür nur auf dem Bildschirm zur Anzeige bringen. Zur Anzeige des Programms verwenden wir das LIST Kommando.
Mit den Cursortasten kann der Cursor auf dem Bildschirm in beliebiger Richtung bewegt werden; man kann das Ändern oder Löschen von Zeichen in Cursorposition ausführen oder vor dem Zeichen in Cursorposition weitere einfügen.
Folgende Tasten werden beim Editieren von Programmen verwendet:

[→]:	Bewegt den Cursor um ein Zeichen nach rechts
[←]:	Bewegt den Cursor um ein Zeichen nach links
[↑]:	Bewegt den Cursor um eine Zeile aufwärts
[↓]:	Bewegt den Cursor um eine Zeile abwärts
[INST]:	Bewegt das Zeichen in Cursorposition und alle rechts davon stehenden Zeichen um eine Position nach rechts, in Cursorposition wird eine Leerstelle eingefügt. Zum Einfügen mehrerer Zeichen drückt man die [INST] Taste entsprechend mehrmals hintereinander. Wenn beim Einfügen von Leerstellen mit der [INST]-Taste das Ende einer Programmzeile das rechte Ende des Bildschirms erreicht, kann man keine weiteren Leerstellen einfügen. In diesem Falle die [CR]-Taste drücken und den LIST-Befehl ausführen. Das neue Programm-Listing enthält nun im Anschluß an die betreffende Zeile eine Reihe Leerstellen, so daß man weitere Leerstellen einfügen kann.
[DEL]:	Löscht das Zeichen in Cursorposition und bewegt alle Zeichen der Zeile rechts vom Cursor um eine Stelle nach links.
[SHIFT] + [INST] (CLR):	Löscht den Bildschirm. Mit [SHIFT] + [INST] ist gemeint: Die [INST]-Taste drücken, während man die [SHIFT]-Taste gedrückt hält.
[SHIFT] + [DEL] (HOME):	Bewegt den Cursor in die linke obere Ecke des Bildschirms.

Nehmen wir einmal an, daß wir das im Abschnitt 3.3 gezeigte Programm wie folgt eingegeben haben:

```
10 CLS
20 PRINT "BEISTIELPROGRAMM"'
30 PINTT "MZ-800"
40 END
```

Zum Editieren des obigen Programms muß zunächst das Programm-Listing auf dem Bildschirm angezeigt werden. Falls es zur Zeit nicht angezeigt wird, bringt man es durch Ausführung des LIST Kommandos zur Anzeige.

(1) Ersetzen eines Buchstabens

BEISTIEL in Zeile 20 sollte stattdessen BEISPIEL heißen. Bewegen Sie zunächst mit den Cursorsteuertasten den Cursor auf den falsch geschriebenen Buchstaben T und drücken Sie [P]. Wenn der Buchstabenaustausch korrekt vorgenommen ist, drückt man die [CR]-Taste, woraufhin der Curspr am Anfang von Zeile 30 positioniert wird.

(2) Einfügen eines Buchstabens

Nun bewegen wir den Cursor auf das I von PINTT und drücken die [INST]-Taste. Zum Einfügen eines R zwischen P und I drücken Sie nun [R].

(3) Löschen eines Buchstabens

Setzen Sie den Cursor auf eins der beiden T bei PRINTT und drücken Sie die [DEL]-Taste zum Löschen des Zeichens. Nun die [CR]-Taste drücken. Der Cursor wird zum Anfang von Zeile 40 bewegt.

(4) Hinzufügen von Zeilen

Eine neue Zeile kann an jeder beliebigen Stelle des Programms eingefügt werden. Nehmen wir einmal an, zwischen Zeile 10 und Zeile 20 soll eine Zeile eingefügt werden. Bewegen Sie hierzu den Cursor an den Anfang der Zeile unter Zeile 40. Nun schreiben Sie "15 REM Editier-Beispiel" [CR].

```
10 CLS
20 PRINT "BEISPIELPROGRAMM"
30 PRINT "MZ-800"
40 END
15 REM Editier-Beispiel
```

Zwischen Zeile 10 und Zeile 20 können Zeilen mit Nummern von 11 bis 19 eingefügt werden. Nun durch Eingabe des CLS-Befehls und des LIST Kommandos hintereinander sicherstellen, daß die neue Zeile zwischen den Zeilen 10 und 20 eingefügt wird.

```
LIST
10 CLS
15 REM Editier-Beispiel
20 PRINT "BEISPIELPROGRAMM"
30 PRINT "MZ-800"
40 END
```

(5) Löschen von Zeilen

Das DELETE Kommando kann man zum Löschen von beliebigen Programmzeilen verwenden. Löschen wir einmal die Zeilen 15 und 20. Hierzu folgendes eingeben:

DELETE 15—20 [CR]

Geben Sie nun LIST ein. Das Programm-Listing sollte wie folgt aussehen:

```
10 CLS
30 PRINT "MZ-800"
40 END
```

(6) Neunumerierung

Durch Eingabe des RENUM Befehls werden die Zeilennummern neu numeriert, und zwar jeweils mit einer Zunahme um den spezifizierten Wert (falls dieser nicht spezifiziert wird, so erfolgt automatisch Zunahme um 10).

```
RENUM [CR]
LIST [CR]
10 CLS
20 PRINT "MZ-800"
30 END
```

(7) AUTO Befehl

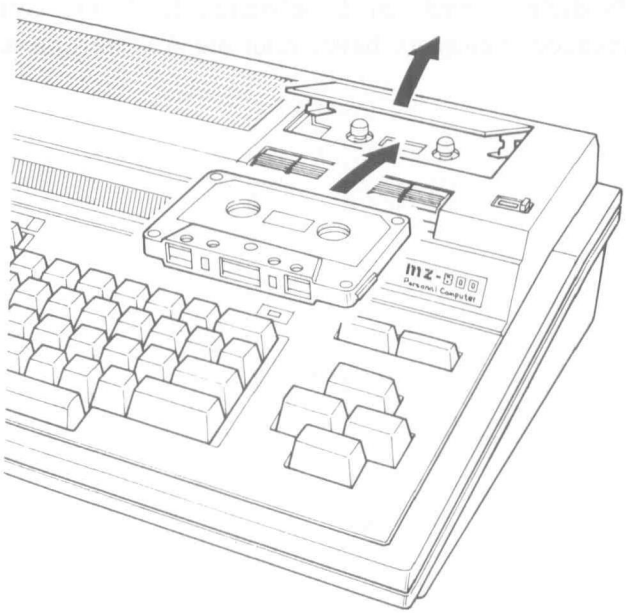
Der AUTO Befehl ist praktisch zur automatischen Erzeugung von Zeilennummern mit jeweiliger Zunahme um den spezifizierten Wert. Für weitere Einzelheiten siehe Kapitel 6. Auf jeden Fall stets die [CR]-Taste drücken, nachdem man eine Zeile fertig editiert hat, da andernfalls die Zeile nicht so verändert wird, wie es auf dem Bildschirm erscheint. Desgleichen den Cursor auf jeden Fall in die Zeile unterhalb der letzten Programmzeile bewegen, bevor man die Ausführung des Programms startet.

3.5 Speichern eines Programms

Wenn man den Netzschalter betätigt, wird das gerade eingegebene Programm zerstört bzw. gelöscht. Um das Programm dauerhaft zu sichern, muß man es auf einem externen Datenträger, wie zum Beispiel einer Magnetbandcassette speichern. Nachstehend ist die Verfahrensweise zum Speichern eines Programms auf Cassette beschrieben.

*** Verwendung einer neuen Cassette**

1) Das Cassettenfach öffnen und die Cassette einlegen. Zum Rückstellen des Bandzählwerks auf "000" die Bandzählwerk-Rückstelltaste drücken.



2) Schreiben Sie den Befehl zum Sichern des Programms im folgenden Format:

   SAVE "CMT:TEST" [CR]

   Dieser Befehl weist den Computer an, das im Speicher stehende Programm unter dem Namen "TEST"auf der Cassette im Cassettenlaufwerk (ausgewiesen durch den Namen CMT:) zu speichern.
3) Wenn die Meldung "↓RECORD.PLAY" angezeigt wird, drücken Sie die Aufnahmetaste (RECORD).
4) Wenn die Meldung "Ready" auf dem Bildschirm erscheint und der Bandlauf stoppt, drücken Sie die [STOP]-Taste. Es ist empfehlenswert, den Programmnamen "TEST" und den Zählerstand auf dem Cassettenetikett oder in ein eigens dafür vorgesehenes Verzeichnis einzutragen.

*** Verwendung einer Cassette, die Programme enthält:**

In diesem Falle muß der Zählerstand vom Ende des letzten Programms bekannt sein, andernfalls können auf dem Band enthaltene Programme zerstört werden.

1) Die Cassette ins Cassettenlaufwerk einlegen und das Band durch Drücken der Rückspultaste (REWIND) zurückspulen.
2) Zum Rückstellen des Bandes auf "000" die Bandzählwerk-Rückstelltaste drücken.
3) Die Schnellvorlauf-Taste FFWD drücken. Wenn der Zählwerkstand vom Ende des letzten Programms fast erreicht ist, den Bandlauf durch Drücken der [STOP]-Taste stoppen.
4) Nun die Wiedergabetaste (PLAY) drücken. Wenn der Zählwerkstand vom Ende des letzten Programms zzgl. 2 oder 3 erreicht ist, die [STOP]-Taste drücken.

5) Für eine neue Cassette die Schritte 2 bis 4 des oben beschriebenen Verfahrens ausführen.
6) Nach Speicherung des Programms den Zählerwert notieren und das Band zusätzlich 2 oder 3 Umdrehungen laufen lassen, dann die Stoptaste (STOP) drücken. Wenn Programme auf einer Cassette gespeichert werden, auf der sich schon Programme befinden, dann könnten diese Programme gelöscht werden, wenn dieses Verfahren angewandt wird. Deshalb sollte immer eine neue Cassette für Programmspeicherung verwendet werden.

3.6 Laden eines Programms

Die zusammen mit dem Computer gelierferte Cassette enthält ein Demonstrationsprogramm. Es können jedoch ebenfalls auf dem Markt erhältliche Programme verwendet werden. Zur Anwendung dieser Programme sowie der von Ihnen erstellten, müssen diese von der Cassette in den Computerspeicher geladen werden. Im Folgenden wird die Ladung eines Programms beschrieben.

1) Die das Programm enthaltende Cassette ins Cassettenlaufwerk einlegen. Falls erforderlich, das Band zurückspulen.
2) Zum Einlesen des Programms in den Arbeitsspeicher des Computers den folgenden Befehl eingeben:

   LOAD "CMT: Programmname"

   Um z.B. das Programm "TEST" zu laden, schreiben wir:

   LOAD "CMT:TEST"
3) Wenn die Meldung " ⬇ PLAY" erscheint, die Wiedergabetaste (PLAY) des Cassettenlaufwerks drücken.
4) Wenn der Bandlauf stoppt, die [STOP]-Taste drücken.
5) Zur Ausführung des Programms gibt man RUN ein, wenn die Meldung "Ready" auf dem Bildschirm erscheint.

```
Ready
RUN [CR]
```

Kapitel 4 Tastatur und Cassettenlaufwerk

4.1 Die Tastatur

Die Tastatur des MZ-800 arbeitet stets in einer der folgenden Betriebsarten:

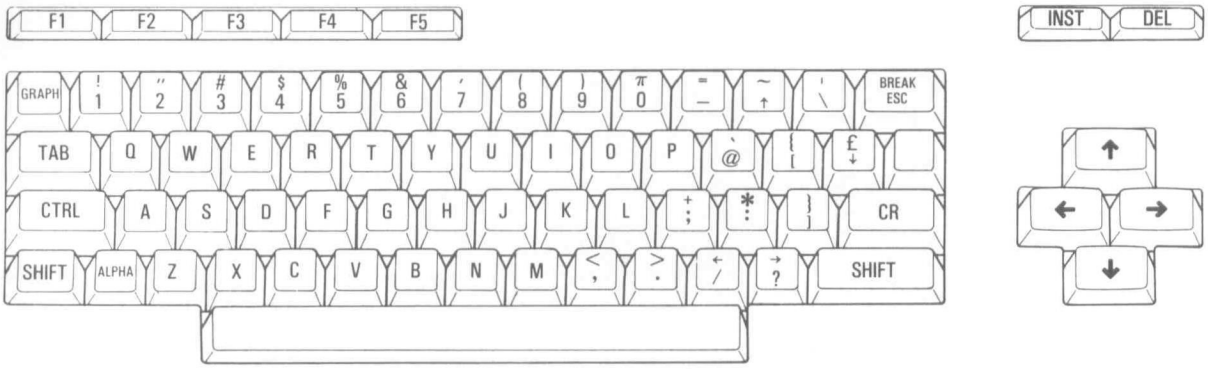
* Normal-Betriebsart: Wird normalerweise zur Eingabe von Buchstaben, Zahlen und Symbolen verwendet. Diese Betriebsart wird automatisch vorgegeben, wenn der BASIC Interpreter gestartet oder eine Rückstellung vorgenommen wird.

* Umschaltsperre: In dieser Betriebsart befinden sich alle Tasten mit Ausnahme der Funktionstasten [F1] bis [F5] im Umschaltzustand. Durch Drücken von [SHIFT] + [ALPHA] wird auf diese Betriebsart umgeschaltet. Durch nochmaliges Drücken von [SHIFT] + [ALPHA] wird die Umschaltsperre wieder aufgehoben.

* Grafik-Betriebsart: Dient zur Eingabe von Grafik-Spezialzeichen. Zur optischen Unterscheidung der zur Zeit aktivierten Tastatur-Betriebsart werden drei verschiedene Ausführungen von Cursor angezeigt:
  ■ : Cursor für Normal-Betriebsart
  ● : Cursor in der Umschaltsperre-Betriebsart
  _ : Cursor für Grafik-Betriebsart

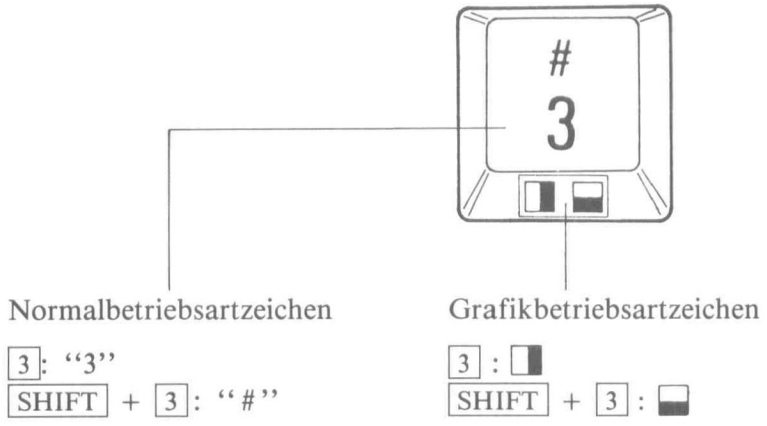
4.1.2 Tasten

Die Tastatur des MZ-800 enthält eine Vielzahl von Tasten, deren Funktionen nachstehend beschrieben sind.

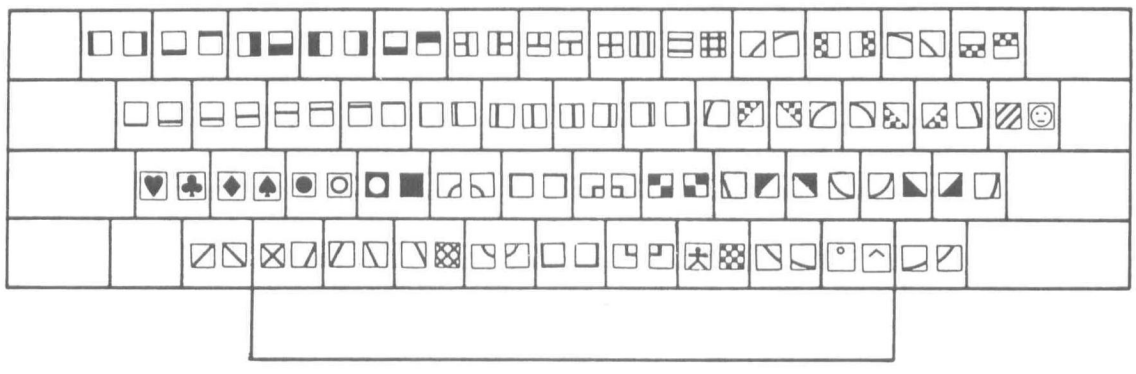


(1) Zeichen-Tasten

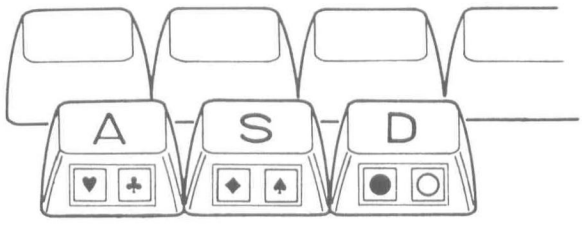
Zur Eingabe von Buchstaben, Zahlen und grafischen Symbolen verwendete Tasten. Die Tastenkappen bestimmter Zeichen-Tasten sind mit zwei verschiedenen Zeichen markiert. Diese Zeichen werden eingegeben, wenn man die jeweilige Taste in der einen oder der anderen Betriebsart betätigt.



In der Normal-Betriebsart: Durch Drücken einer Zeichen-Taste wird der auf der Taste dargestellte Buchstabe in Großschreibung bzw. das im unteren Tail der Tastenkappe aufgedruckte Zeichen eingegeben. Wenn man die Taste zusammen mit der SHIFT-Taste drückt, wird der auf der Taste dargestellte Buchstabe in Kleinschreibung oder das im oberen Teil der Tastenkappe aufgedruckte Zeichen eingegeben.
In der Grafik-Betriebsart: Durch Drücken einer Zeichen-Taste wird das links auf der Taste dargestellte Zeichen eingegeben, wenn man die Zeichen-Taste zusammen mit der SHIFT-Taste drückt, wird das rechts auf der Taste dargestellte Zeichen eingegeben.

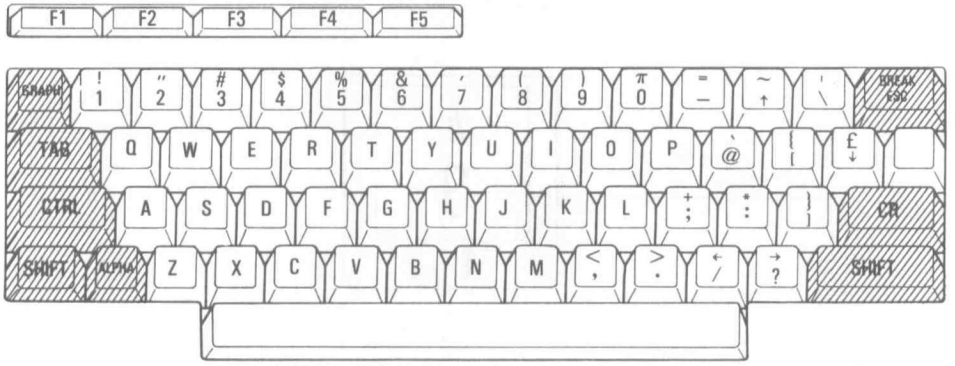


Diese Grafik-Symbole sind nicht unmittelbar auf den Tasten aufgedruckt. Ein Satz Klebeetiketten mit aufgedruckten Grafiksymbolen gehört jedoch zur Standardausstattung des MZ-800. Es ist vielleicht praktisch, die Etiketten wie in der obigen Abbildung dargestellt, an der Frontseite der Tasten anzubringen.



(2) Spezialtasten

Diese Tasten dienen zur Steuerung des Computers und zur Bestimmung der Eingabe-Betriebsart der Zeichen-Tasten. Die Spezialtasten sind in der nachstehende Abbildung schraffiert dargestellt.



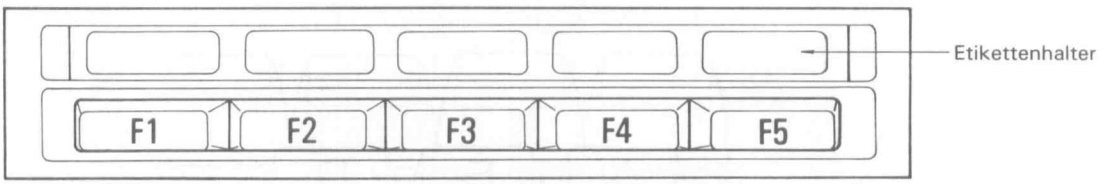
Die Spezialtasten haben folgende Funktionen.

CR	: Diese Taste dient zur Eingabe der den Cursor enthaltenden Zeile in den Computer. Die eingegebenen Zeichen erscheinen zwar unmittelbar auf dem Bildschirm, sie werden vom Computer jedoch solange ignoriert, bis man die CR-Taste drückt.
SHIFT	: Durch Drücken dieser Taste werden die Tasten der Tastatur in den Umschalte-Zustand versetzt.
GRAPH	: Durch Drücken dieser Taste wird für die Tastatur die Graphik-Betriebsart aktiviert.
ALPHA	: Durch Drücken dieser Taste wird die Normal-Betriebsart der Tastatur wiederhergestellt.
BREAK/ESC	: Diese Taste dient zur Eingabe von ESC-Codes (Umschaltsteuerzeichen) und zum Anhalten des Programmes ohne Abbruch.
SHIFT + BREAK/ESC	: Diese Tastenkombination dient zum Stoppen der Ausführung eines Programms oder zum Stoppen des Cassettenlaufwerkbetriebs.
TAB	: Zur Weiterbewegung des Cursors bis zum nächsten Tabstop auf dem Bildschirm. Tabstops werden im Abstand von je 10 Zeichen gesetzt.
CTRL	: Durch Drücken einer Zeichen-Taste, während man diese Taste gedrückt hält, wird ein Steuerzeichen eingegeben. Für weitere Einzelheiten über Steuerzeichen, siehe Anhang D.

(3) Editier-Tasten

Die Editier-Tasten dienen zum Einfügen, Ändern oder Löschen von Programmteilen. Diese Tasten befinden sich auf der rechten Seite der Tastatur. Für weitere Einzelheiten über die Funktionen dieser Tasten, siehe Abschnitt 3.4.

(4) Vom Benutzer definierbare Tasten



Unmittelbar nachdem der BASIC Interpreter aktiviert wurde, werden den definierbaren Tasten folgende Funktionen zugewiesen. Die Funktionen dieser Tasten können unter Verwendung von DEF KEY Anweisung nach Wunsch verändert werden. Siehe Kapitel 6.

[F1] : "RUN␣␣␣" + CHR$(13)
[F2] : "LIST␣"
[F3] : "AUTO␣"
[F4] : "RENUM␣"
[F5] : "COLOR␣"
[SHIFT] + [F1] : "CHR$("
[SHIFT] + [F2] : "DEF␣KEY("
[SHIFT] + [F3] : "CONT"
[SHIFT] + [F4] : "SAVE␣␣"
[SHIFT] + [F5] : "LOAD␣␣"

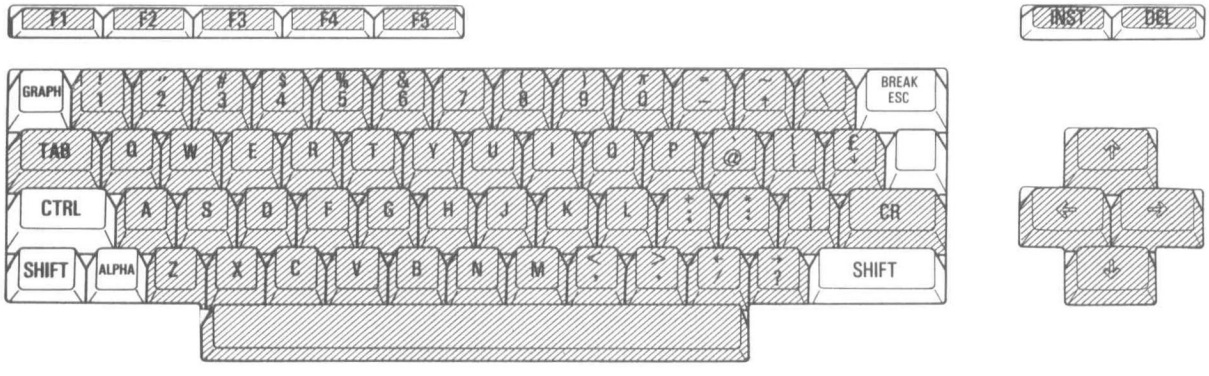
Hinweis:
CHR$(13) ist das Steuerzeichen für die [CR]-Taste und "␣" steht jeweils für eine Leerstelle.

* Anbringen von Etiketten für die vom Benutzer definierbaren Tasten.

Wir empfehlen das Einsetzen der mitgelieferten Etiketten für die definierbaren Tasten in die Etikettenhalter, die sich unmittelbar über den definierbaren Tasten befinden. Zum Einsetzen der Etiketten wird die Etiketten-Klarsichtabdeckung geöffnet, indem man sie hochzieht.

(5) Automatische Wiederholungs-Funktion

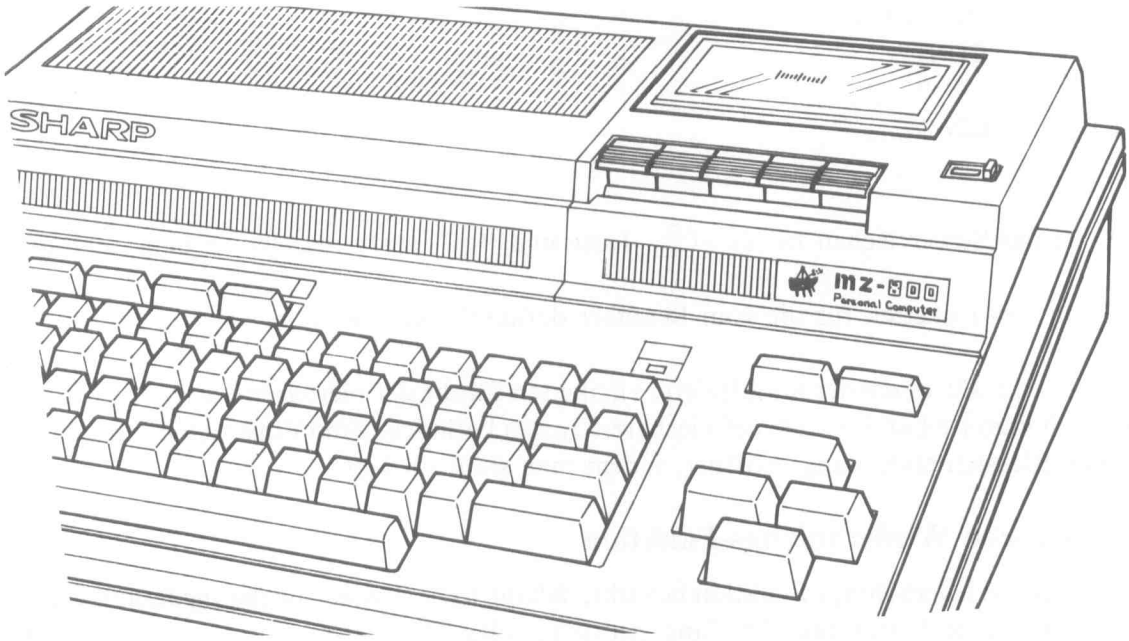
Die automatische Wiederholungs-Funktion bewirkt, daß die letzte Tasteneingabe wiederholt wird, wenn man diese Taste eine bestimmte Zeit lang gedrückt hält. Die Tasten, für die die automatische Wiederholungs-Funktion wirksam ist, sind in der nachstehenden Zeichnung schraffiert dargestellt.



4.2 Cassettenlaufwerk

(1) Hardware

Der MZ-800 ist mit einem Cassettenlaufwerk ausgestattet.



Die Funktionen der einzelnen Tasten sind wie folgt:

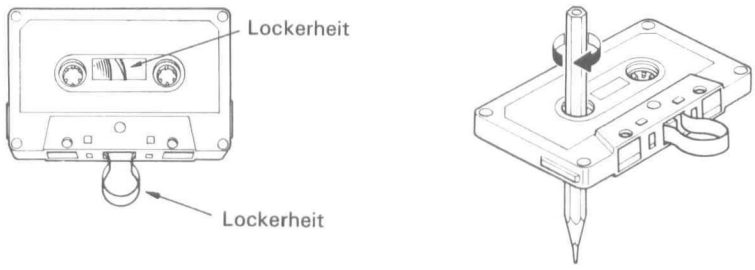
PLAY	:Durch Drücken dieser Taste läuft das Band im Wiedergabebetrieb in Vorwärtsrichtung; dient zum Laden eines Programms von der Cassette in den Arbeitsspeicher.
RECORD	:Durch Drücken dieser Taste zusammen mit der Wiedergabetaste (PLAY) wird das im Speicher befindliche Programm auf Cassette gespeichert.
FFWD	:Durch Drücken dieser Taste erfolgt schneller Vorlauf des Bandes.
REWIND	:Betätigung dieser Taste spult das Band zurück.
STOP/EJECT	:Zum Stoppen des Bandlaufs diese Taste drücken. Wenn man die Taste drückt, während das Band nicht läuft, wird die Cassette ausgeworfen.

Bandzählwerk-Rückstelltaste: Zum Rückstellen des Bandzählwerks auf "000".

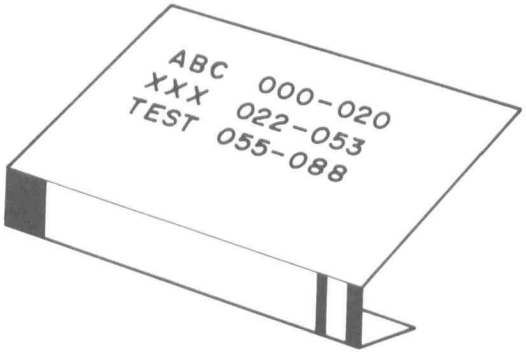
Hinweis:
Wenn das Bandende erreicht ist, wird die Raststellung der FFWD- und der REWIND-Taste nicht automatisch gelöst. Auf jeden Fall am Bandende die STOP-Taste drücken.

(2) Hinweise zum Umgang mit der Cassette

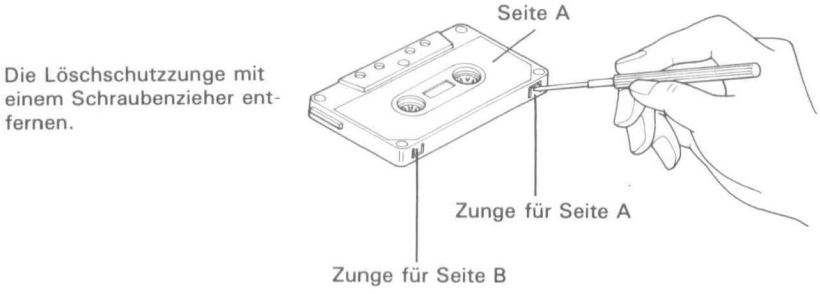
* Mit dem MZ-800 können alle handelsüblichen Cassetten verwendet werden. Wir empfehlen jedoch, nur Markencassetten von zuverlässigen Herstellern zu verwenden.
* Keine Normalbänder verwenden.
* Den Gebrauch von C-120 Cassetten möglichst vermeiden.
* Wir empfehlen die Verwendung von C-60 oder kürzeren Cassetten.
* Vor dem Einlegen der Cassette auf jeden Fall lockere Bandstellen mit einem Bleistift o.dgl aufspulen.



Es empfiehlt sich, nach dem Speichern den Programmnamen und den Zählerstand vom Anfang und Ende des aufgenommenen Programms zu notieren.



* Cassetten nicht in der Nähe eines Fernsehgerätes, Lautsprechers oder sonstiger Magnetfelder erzeugender Gegenstände aufbewahren.
* Schutz von Programmen/Daten vor unbeabsichtigter Löschung
  Um die unbeabsichtigte Löschung von Daten zu verhindern, bricht man die Löschschutzzungen vom Cassettengehäuse mit einem Schraubenzieher oder einem ähnlichen Gegenstand aus. Dadurch wird es unmöglich gemacht, die Aufnahmetaste [RECORD] herunterzudrücken, so daß wertvolle Daten nicht versehentlich gelöscht werden können.



Kapitel 5 Programmier-Konzepte

In diesem Kapitel befassen wir uns mit den fundamentalen Konzepten zur Programmierung in BASIC mit dem MZ-800. Die in diesem Kapitel enthaltenden Informationen sind von Bedeutung, um das volle Potential von BASIC nutzen zu können.

5.1 Mehrere Anweisungen enthaltende Zeilen und Zeilennummern

Wie schon im Kapitel 3 ausgeführt, enthält ein Programm eine oder mehrere Befehlszeilen. In den Beispielen von Kapitel 3 ist nur jeweils ein Befehl pro Zeile enthalten; in einer Programmzeile können jedoch auch zwei oder mehr Befehle stehen, vorausgesetzt, daß diese durch Doppelpunkt (:) voneinander getrennt sind.

Beispiel:

```
10 CLS:PRINT "MEHRERE BEFEHLE IN EINER ZEILE": END
```

Jede Programmzeile beginnt mit einer Zeilennummer. Die Zeilennummern müssen im Bereich von 1 bis 65535 liegen. Man muß die Zeilen nicht in durchgehender Reihenfolge numerieren, im Gegenteil, es ist sogar empfehlenswert, die Zeilen jeweils mit Zehnerschritten zu numerieren, so daß zusätzliche Zeilen während des Editierens eingefügt werden können.

5.2 Numerische Daten und Zeichenfolge-Daten

Die vom Computer verarbeiteten Daten werden ganz allgemein in numerische Daten und Zeichenfolge-Daten (Stringdaten) aufgeteilt. Dabei verkörpern numerische Daten bestimmte Größen oder Mengen, wohingegen Zeichenfolge-Daten aus Zeichen bestehen.

(1) Numerische Daten

Mit MZ-800 BASIC ist die Bearbeitung von numerischen Daten sowohl in dezimaler als auch in hexadezimaler Schreibweise möglich. Ungeachtet der gewählten Schreibweise werden alle Daten jedoch vom Computer in Binärform umgewandelt; in diesem Format werden sie gespeichert und Rechnungen damit vorgenommen.
In der hexadezimalen Schreibweise werden die Ziffern von 0 bis 9 verwendet und daüber hinaus die Zeichen von A bis F zur Darstellung der Werte von 10 bis 15.
Mit diesem System nimmt die Anzahl der Stellen, die zum Ausdrücken von Zahlen erforderlich ist, jeweils basierend auf dem Faktor 16 um eins zu.
Zur Darstellung von Hexadezimalzahlen wird diesen wie nachstehend dargestellt, das Dollarzeichen "$" vorangestellt.

$$\$41 = 4 \times 16^1 + 1 \times 16^0 = 65$$
$$\$FA = 15 \times 16^1 + 10 \times 15^0 = 250$$

Für Hexadizimalzahlen größer als $7FFF werden Komplemente als Ergebnis erzielt. Zum Beispiel ist der für $8000 erzielte Wert −32768 und der für $FFFF ist −1.

(2) Zeichenfolge-Daten

Alle Zeichen werden intern als numerischer Code, d.h. in Form von Zahlen dargestellt. Dieser numerische Code basiert auf dem ASCII Code-System. Mit dem ASCII Code-System werden die Zeichen durch Zahlen von 0 bis 255 bzw. $00 bis $FF dargestellt. So wird z.B. das Zeichen "A" im ASCII Code als 65 oder $41 dargestellt.

5.3 Konstanten

(1) Numerische Konstanten

Numerische Konstanten sind positive oder negative Zahlen. Sie werden entweder in gewöhnlicher Form oder in Exponentialform dargestellt. Numerische Konstanten müssen innerhalb eines Bereiches von 10^{-38} bis 10^{38} liegen (1E−38 bis 1.7014118E+38), und die max. Anzahl wesentlicher Ziffern beträgt 8. Wenn der Wert einer Konstanten außerhalb des Bereiches liegt, ist die Richtigkeit des Ergebnisses nicht gewährleistet.
Gewöhnliche Ganzzahlen und Dezimalzahlen werden in der folgenden Normalform dargestellt.

```
123
-123.4
+12
```

Bei positiven Zahlen kann das "+" auch ausgelassen werden. Extrem große oder extrem kleine Zahlen werden in Exponentialform dargestellt. In diesem Format wird die Zahl als eine Kombination einer Zahl (Mantisse), E und einer weiteren Zahl (Exponent) dargestellt. Das Zeichen "E" kennzeichnet die Darstellung in Exponentialform. Z.B. steht

```
1.23E+2
```

für $1.23 \times 10 \times 10 = 123$

```
-1.2E-1
```

steht für $-1.2 \div 10 = -0.12$

Bei positiven Zahlen kann das "+" auch ausgelassen werden. Die Mantisse muß kleiner als 10 und größer als −10 sein und der Exponent eine Ganzzahl von −38 bis +38.

(2) Zeichenfolge-Konstanten

Eine Zeichenfolge-Konstante ist eine in (" ") eingeschlossene Zeichenfolge. Die max. Anzahl Zeichen in einer Zeichenfolge-Konstanten richtet sich nach der effektiven Zeilenlänge, die max. zulässige Gesamtanzahl Zeichen einer Zeichenfolge beträgt jedoch 255. Beispiele von Zeichenfolge-Konstanten sind nachstehend aufgeführt.

```
"ABC"
"12345"
"MZ-800"
```

Hinweis:
In DATA Anweisungen sind Anführungszeichen nur erforderlich, wenn ein Komma im Datum enthalten ist. (Siehe Kapitel 6.)

Beispiel: 200 DATA 21, 24.5, "24.5", BASIC

5.4 Variablen

Variablen sind Speicherplätze, die dazu dienen, während der Programmausführung bestimmte Werte zu enthalten. Beim Schreiben eines Programms muß jede verwendete Variable einen eindeutigen Namen erhalten. Die Werte der Variablen können während der Programmausführung beliebig verändert werden, wobei jedoch kein Wechsel im Datentyp vorgenommen werden kann.
MZ-800 BASIC kann drei verschiedene Typen von Variablen handhaben: numerische Variablen, Zeichenfolge-Variablen und Systemvariablen.

(1) Numerische Variablen

Numerische Variablen können nur numerische Daten enthalten. Der Name jeder Variablen kann aus einer beliebigen Anzahl Zeichen bestehen, aber nur die ersten zwei Zeichen dienen zur Kennzeichnung der Variablen. So sind z.B. AB und XYZ verschiedene Variablen, aber ABC und ABD werden als die gleiche Variable behandelt.
Kleinbuchstaben dürfen nicht für Variablen verwendet werden.
Das erste Zeichen muß ein Buchstabe von A bis Z sein, das zweite und die folgenden können entweder Buchstaben des Alphabets oder Zahlen sein, Spezialzeichen wie z.B. @ und * können jedoch nicht verwendet werden. Ferner dürfen keine reservierten Wörter (siehe Anhang F) als Variablennamen verwendet werden. Z.B. PRINT und C können nicht als Variablennamen verwendet werden.
Grundsätzlich enthält jede numerische Variable den Wert 0, bis ihr ein anderer Wert zugewiesen wird.

(2) Zeichenfolge-Variablen (Stringvariablen)

Eine Zeichenfolge-Variable kann nur Zeichenfolge-Daten enthalten. Ihr Name wird in der gleichen Weise gebildet wie der Name einer numerischen Variable, wobei dem festgelegten Namen stets ein Dollarzeichen ($) nachgestellt wird.
Jede Zeichenfolge-Variable kann max. 255 Zeichenfolge-Daten enthalten; solange der Variablen noch keine Zeichenfolge-Daten zugewiesen sind, enthält sie null Zeichen.

(3) Systemvariablen

Es gibt bestimmte Variablen, sog. Systemvariablen, die vom BASIC Interpreter definiert und verwendet werden. In der folgenden Tabelle sind diese Systemvariablen aufgeführt.

Systemvariable	Erklärung
TI$	Enthält eine sechsstellige Zahl, die basierend auf der eingebauten Uhr im 24-Stundenformat die Uhrzeit darstellt. Z.B entspricht der Wert "192035" der Uhrzeit 19:20:35 auf 24-Stunden-Basis. Bei Einschalten des Gerätes wird die Uhr stets auf 00:00:00 gestellt.
SIZE	Zur Anzeige des freien Speicherbereichs, der für BASIC Programme und Daten zur Verfügung steht.
ERN	Bei Auftreten des Fehlers enthält diese Variable die zugehörige Fehlernummer.
ERL	Bei Auftreten eines Fehlers enthält diese Variable die Zeilennummer, in der der Fehler auftrat.
CSRH	Enthält die gegenwärtige Spaltenposition des Cursors $0 \leqq CSRH \leqq 39$ (Betriebsart 40-Spalten-Bildschirm) $0 \leqq CSRH \leqq 79$ (Betriebsart 80-Spalten-Bildschirm)
CSRV	Enthält die Nummer der Zeile, in der der Cursor sich zur Zeit befindet. $0 \leqq CSRV \leqq 24$
POSH	Enthält die X-Koordinate des Graphikpositions-Zeigers. $-16384 \leqq POSH \leqq 16383$
POSV	Enthält die Y-Koordinate des Graphikpositions-Zeigers. $-16384 \leqq POSV \leqq 16383$

5.5 Feldvariablen

Unter einer Feldvariable versteht man die Zusammenfassung mehrerer Variablen gleichen Typs unter einem gemeinsamen Namen. Jedes Elements wird einerseits gekennzeichnet durch eine Zeichenfolge, die grundsätzlich in der gleichen Weise aufgebaut ist wie ein gewöhnlicher Variablenname, und andererseits durch einen nachgestellten, in Klammern eingeschlossenen Index, z.B. A(x) oder B$(x,y). Eine Feldvariable mit einem Index (z.B. A(x), B$(1) oder P(100)) nennen wir eindimensionale Feldvariable und solche mit zwei Indizes (z.B. A(x,y), B$(1,3) oder P(50,25)) nennen wir zweidimensionale Feldvariable. Bevor man Feldvariablen in einem Programm verwenden kann, muß der gemeinsame Name und die Anzahl der enthaltenen Variablen vereinbart (d.h. festgelegt) werden. Für weitere Einzelheiten hierzu siehe Erklärung der DIM Anweisung in Kapitel 6.

• Hinweis über Rechenfehler

Bei Verwendung von Computern muß man sich stets darüber im klaren sein, daß Rechenfehler auftreten können. Der Grund dafür ist, daß zwar das Ausmaß des Rechenfehlers durch Erhöhung der Stellenanzahl verarbeitbarer numerischer Daten reduziert werden kann, jedoch kann kein Computer unbegrenzt viel Stellen verarbeiten.
Darüberhinaus ist zu berücksichtigen, daß die Rechnung um so länger dauert, je mehr Stellen die Zahlen haben. Es ist daher wichtig, daß man sich mögliche Fehlerquellen vergegenwärtigt und Programme so schreibt, daß Fehler minimiert werden. (Zum Beispiel sollte man die Folge "5 * 6/3 anstelle von 5/3 * 6" verwenden.)

Bei der Ausführung von Rechnungen in BASIC (1Z016) für den MZ-800 bitte folgendes berücksichtigen.

(1) Rundungsfehler
Rundungsfehler sind Fehler, die auftreten, wenn die Anzahl der Stellen rechts vom Dezimalpunkt die Anzahl der Stellen, die effektiv gehandhabt werden können, überschreitet. Wenn z.B. die Zahl 2/3 berechnet wird, ist das Ergebnis 0.666666 . . . . (wobei die Reihe der Sechsen unbegrenzt ist). Wenn die Anzahl der wirksamen Stellen jedoch 8 ist, wird das Ergebnis auf 0.66666667 gerundet.

(2) Fehler nach Umwandlung in die Binärform
Obwohl Zahlen normalerweise im Dezimalformat eingegeben werden, erfolgt eine interne Umwandlung ins Binärformt zwecks Berechnung. Demzufolge kann nach der Umwandlung eine Binärzahl mit unbegrenzter Stellenanzahl auftreten, selbst wenn die ursprüngliche Dezimalzahl nur wenige Stellen hat. Wenn man z.B. die Dezimalzahl 0.1 ins Binärformat umwandelt, ist das Ergebnis 0.00011001100 . . . . Da dies gerundet werden muß, um die Rechnung ausführen zu können, entsteht eine bestimmte Fehlergröße.

(3) Zunahme des relativen Fehlers aufgrund von Subtraktion
Wenn man eine Zahl von einer anderen abzieht, ist die relative Fehlergröße im Ergebnis größer als in den ursprünglichen Zahlen. Dies wird durch das nachstehende Beispiel veranschaulicht, in dem die Fehler enthaltenden Stellen mit einem Punkt (.) markiert sind. Ein Fehler von ±1 in der Zahl 100012 entspricht einem Fehlerprozentsatz von ca. 0.001%; der relative Fehler ist nach der Subtraktion jedoch erheblich größer, da 11 ±1 einem relativen Fehler von ca. 10% entspricht.

$$\begin{array}{r} 100012 \\ -)\underline{100001} \\ 11 \end{array}$$

(4) Fehler aufgrund von Näherungsrechnung
Mit einem Computer werden Exponentiation, trigonometrische und Logarithmus-Rechnungen unter Verwendung des Näherungsverfahrens ausgeführt; demzufolge tritt bei der Ausführung derartiger Berechnungen ein bestimmter Näherungsfehler auf.

5.6 Ausdrücke

Ein Ausdruck ist eine Kombination von Variablen und Konstanten, die durch Operatoren miteinander verknüpft sind. Operatoren sind Symbole zur Darstellung von mathematischen oder logischen Operationen, die mit gegebenen Werten ausgeführt werden sollen. Mit MZ-800 BASIC werden die folgenden Arten von Ausdrücken verarbeitet.

* Arithmetische Ausdrücke
* Zeichenfolgen verbindende Ausdrücke
* Vergleichsausdrücke
* Logische Ausdrücke

(1) Arithmetische Ausdrücke

Ein arithmetischer Ausdruck besteht aus arithmetischen Operatoren, numerischen Konstanten, numerischen Variablen und numerischen Funktionen. Er produziert einen numerischen Wert als das Ergebnis der durch den Operator vorgeschriebenen Operation. (Die numerischen Funktionen werden später in diesem Kapitel erklärt.) In der nachstehenden Tabelle sind die arithmetischen Operatoren in der Reihenfolge ihrer Rechenpriorität aufgeführt.

Arithmetischer Operator	Operation	Beispiel
( )	Von Klammern eingeschlossene Operationen haben höchste Priorität	(X+Y)
↑	Exponentenbildung	X↑Y
−	Umwandlung des Vorzeichens eines Wertes	−X
*,/	Multiplikation, Division	X*Y, X/Y
+,−	Addition, Subtraktion	X+Y, X−Y

Wenn ein arithmetischer Ausdruck Operationen gleicher Priorität enthält, werden diese der Reihe nach von links nach rechts ausgeführt.

(2) Zeichenfolgen verbindende Ausdrücke

Zeichenfolgen verbindende Ausdrücke werden zur Zusammenfassung von 2 oder mehreren Zeichenfolgen-Datenmengen zu einer einzigen verwendet. Ein Zeichenfolgen verbindender Ausdruck enthält Zeichenfolgen-Konstanten, Zeichenfolgen-Variablen, Zeichenfolgen-Funktionen und den Operator "+". (Die Zeichenfolge-Funktion wird später in diesem Kapitel erklärt.)

Beispiele:

```
"ABC" + "DEF" → "ABCDEF"
'A" + "B" + "C" → "ABC"
```

(3) Vergleichsausdrücke

Ein Vergleischsausdruck wird zum Vergleichen von zwei Werten verwendet und gibt je nach dem Ergebnis des Vergleichs den logischen Wert für wahr (−1) oder falsch (0) aus. Das Ergebnis wird dafür verwendet, bezüglich des nachfolgenden Programmablaufs eine Entscheidung zu treffen.Ein Vergleichsausdruck besteht aus Konstanten, Variablen, arithmetischen Ausdrücken, Zeichenfolgen verbindenden Ausdrücken und Vergleichsoperatoren.
Die nachstehende Tabelle enthält eine Aufstellung der Vergleichsoperatoren.

Operator	Vergleich	Beispiel
=	ist gleich	X=Y
<	kleiner als	X<Y
>	größer als	X>Y
<=,=<	kleiner als oder gleich	X<=Y, X=<Y
>=,=>	größer als oder gleich	X>=Y, X=>Y
<>,><	ungleich	X<>Y, X><Y

(4) Logische Ausdrücke

Ein logischer Ausdruck dient zum Ausdrücken der durch die Vergleichsausdrücke angegebenen logischen Verknüpfung von wahren oder falschen Werten (−1) oder (0).
Ein logischer Ausdruck wird durch logische Werte, Vergleichsausdrücke und logische Operatoren gebildet. Die folgende Tabelle enthält die logischen Operatoren.

Operator	Bedeutung	Beispiel
NOT	logische Negation	␣ NOT X
AND	Konjunktion	X ␣ AND Y
OR	logische Summe	X ␣ OR Y
XOR	exklusives ODER	X ␣ XOR Y

Hinweis:
Durch ␣ angezeigte Leerstellen dürfen nicht ausgelassen werden.

5.7 Dateien

Eine Datei enthält ein Programm oder eine Ansammlung von Daten, die von einem Peripheriegerät (z.B. einem Cassettenrecorder) eingegeben oder an ein Peripheriegerät (z.B. Drucker oder Bildschirmanzeige) ausgegeben werden. Eine Datei wird durch den Dateideskriptor gekennzeichnet, der aus dem Dateinamen besteht, wobei diesem Namen der Name des Peripheriegerätes vorangestellt sein kann.

"(Gerätename):(Dateiname)"

Beispiel

CMT:DEMO — Die Datei namens DEMO wird vom Cassettenlaufwerk ein- oder ans Cassettenlaufwerk ausgegeben.

RAM:TEST — Die mit TEST bezeichnete Datei wird in die RAM-Speicherkarte eingegeben oder von ihr ausgegeben.

(1) Dateiname

Ein Dateiname besteht aus max. 16 alphanumerischen Zeichen.

(2) Gerätename

Die folgende Tabelle enthält die von MZ-800 BASIC verwendeten Gerätenamen.

Gerätename	Gerät
CMT:	Cassettenlaufwerk
RAM:	RAM-Speicherkarte (Option)
CRT:	Bildschirmanzeige
LPT:	Drucker
RS1:	Schnittstelle RS-232C
RS2:	

5.8 Funktionen

(1) Numerische Funktionen

Numerische Funktionen wie SIN und COS dienen zur Ausführung von arithmetischen Operationen von gegebenen numerischen Ausdrücken und geben die resultierenden numerischen Werte an. Zur Ausstattung des MZ-800 gehören die folgenden numerischen Funktionen.

ABS(X) — Absolutwert
Zur Ermittlung des Absolutwertes für den numerischen Ausdruck X. In unserem Beispiel wird dieser Wert der Variablen A zugewiesen.
Beispiel: A = ABS(X). Wenn X = 2.9 dann A = 2.9; wenn X = − 5.5 dann A = 5.5.

SGN (X) — Vorzeichen
Je nachdem, ob der numerische Wert größer als, kleiner als oder gleich 0 ist, gibt diese Funktion 1, − 1 oder 0 an.
Beispiel: A = SGN (X). Wenn X = 0.4 dann A = 1; wenn X = − 1.2 dann A = − 1.

INT(X) — Ganzzahl
Zur Angabe der größten Ganzzahl, die kleiner oder gleich X ist.
Beispiel: A = INT(X). Wenn X = 3.87 dann A = 3; wenn X = 0.6 dann A = 0; wenn X = − 3.87 dann A = − 4.

SQR(X) — Quadratwurzel
Zur Angabe der Quadratwurzel von X. Der für X spezifizierte Wert muß größer oder gleich 0 sein.
Beispiel: A = SQR(X). Wenn X = 4 dann A = 2.

EXP(X) — Exponentialfunktion
Gibt den Wert der natürlichen Basis e für die Potenz von X an.
Beispiel: A = EXP(X).

SIN(X)
Gibt den Sinuswert von X an, wobei X ein Winkel im Bogenmaß ist.
Um den Sinus eines Winkels in Grad zu erhalten, verwenden Sie den folgenden Ausdruck.
SIN ($X * \pi/180$)
Beispiel: A = SIN (X)

COS(X)
Zur Angabe des Kosinus von X, wobei X ein Winkel im Bogenmaß ist.
Um den Kosinus eines Winkels in Grad zu erhalten, verwenden Sie den folgenden Ausdruck.
COS ($X * \pi/180$)
Beispiel: A = COS(X)

TAN(X)
Zur Angabe des Tangenswertes von X, wobei X ein Winkel im Bogenmaß ist.
Um den Tangens eines Winkels in Grad zu erhalten, verwenden Sie den folgenden Ausdruck.
TAN ($X * \pi/180$)
Beispiel: A = TAN(X)

ATN(X)
Zur Angabe des Arkustangens von X im Bogenmaß. Der angegenene Wert liegt innerhalb des Bereiches von $-\pi/2$ bis $\pi/2$.
Um den Arkustangens eines Winkels in Grad zu erhalten, verwenden Sie den folgenden Ausdruck.
ATN (X) * 180/π
Beispiel: A = ATN(X)

LOG(X) — Zehnerlogarithmus
Angabe des Zehnerlogarithmus von X ($\log_{10}X$), wobei X größer als 0 sein muß.
Beispiel: A = LOG(X)

LN(X) — Natürlicher Logarithmus
Angabe des natürlichen Logarithmus von X ($\log_e X$), wobei X größer als 0 sein muß.
Beispiel: A = LN(X)

PAI(X) — Konstante für Kreisberechnungen
Angabe des Wertes X mal Pi.
(PAI(1) = π = 3.1415927)
Beispiel: A = PAI(X) oder A = π * X

RAD(X) — Radian (Bogenmaß)
Zur Umwandlung des numerischen Wertes X in Grad in einen Wert im Bogenmaß
Beispiel: A = RAD(X)

(2) Stringfunktionen

Stringfunktionen dienen zur Verarbeitung von Zeichenfolgen. MZ-800 BASIC unterstützt die folgenden Stringfunktionen. In den nachstehenden Beispielen enthält die Stringvariable A$ die Zeichenfolge "ABCDEFG".

LEFT$(x$,n)
x$: Zeichenfolge
n: numerischer Wert (von 0 bis 255)
Angabe einer Zeichenfolge, die aus den n linken Zeichen der Zeichenfolge X$ besteht.
Beispiel: B$ = LEFT$(A$,2) produziert die Zeichenfolge "AB".

MID$(x$,m,n)
x$: Zeichenfolge
m: numerischer Wert von 1 bis 255
n: numerischer Wert von 0 bis 255
Angabe einer Zeichenfolge bestehend aus den n Zeichen im Anschluß an das m-te Zeichen ab Anfang der Zeichenfolge x$.
Beispiel: B$ = MID$(A$,3,3) ergibt die Zeichenfolge "CDE".

RIGHT$(x$,n)
x$: Zeichenfolge
n: numerischer Wert (von 0 bis 255)
Produziert eine Zeichenfolge, die aus den rechten n Zeichen der Zeichenfolge x$ besteht.
Beispiel: B$ = RIGHT$(A$,2) Produziert eine Zeichenfolge, die aus den zwei ganz rechts stehenden Zeichen der Zeichenfolge A$ besteht. B$ entspricht in diesem Falle der Zeichenfolge "FG".

In PRINT Befehlen verwendete Funktionen

TAB(n)

n: numerischer Wert

Bewegt den Cursor zur (n + 1)-ten Zeichenposition ab dem linken Ende der gegenwärtigen Zeile. Diese Funktion wird ignoriert, wenn n kleiner ist als der Spaltenwert der gegenwärtigen Cursorposition.

Beispiel: PRINT "A";TAB(3);"ABC"

```
A       A B C
0 1 2 3 4 5  ← Spaltenposition, wird nicht mit angezeigt.
        └──── Dritte und folgende Zeichen werden angezeigt
```

SPC(n)

n: numerischer Wert

Gibt eine Zeichenfolge bestehend aus mehreren Leerstellen hintereinander an, deren Länge durch n ausgedrückt wird.

Beispiel: PRINT "A";SPC(3);"ABC"

```
A         A B C
0 1 2 3 4 5 6  ← Spaltenposition, wird nicht mit angezeigt.
  └──────── 3-Zeichen-Leerstelle
```

(3) Funktionen zur Umwandlung von numerischen Werten/Zeichenfolgen

Die folgenden Funktionen dienen zur Umwandlung eines numerischen Ausdrucks in eine Zeichenfolge und umgekehrt.

STR$(n)

n: numerischer Wert

Wandelt den numerischen Wert n in eine Zeichenfolge um.

Beispiel: A$ = STR$(– 12)

Der Variablen A$ wird die Zeichenfolge " – 12" zugewiesen.

B$ = STR$(70 * 33)

Der Variablen B$ wird die Zeichenfolge "2310" zugewiesen.

C$ = STR$(1200000 * 5000)

Der Variablen C$ wird die Zeichenfolge "6E + 09" zugewiesen.

Hinweis:

Einer angezeigten oder ausgedruckten positiven Ganzzahl wird eine einzelne Leerstelle vorangestellt, wodurch angezeigt wird, daß das Pluszeichen (+) ausgelassen wurde. Diese Leerstelle entfällt jedoch bei Umwandlung der Ganzzahl in eine Zeichenfolge unter Verwendung der STR$ Funktion.

VAL(x$)

x$: Zeichenfolge

Zur Umwandlung einer Zeichenfolge in einen numerischen Wert.

Beispiel: A = VAL("123")

Die Zeichenfolge "123" wird in den numerischen Wert 123 umgewandelt.

A = VAL ("$FF")

Die Zeichenfolge "$FF" wird in den numerischen Wert 255 umgewandelt.

ASC(x$)

x$: Zeichenfolge

Angabe des numerischen Wertes, der dem ASCII Code des ersten Zeichens der Zeichenfolge x$ entspricht.

Beispiele: X = ASC("A")

X wird ein numerischer Wert von 65 zugewiesen, dieser entspricht dem ASCII Code für Zeichen "A".

Y = ASC("SHARP")

Y wird ein numerischer Wert von 83 zugewiesen, dieser entspricht dem ASCII Code des ersten Zeichens der Zeichenfolge "SHARP".

CHR$(n)

n: numerischer Wert (größer als 32)

Angabe des Zeichens, dessen ASCII Code durch die Ganzzahl n ausgedrückt wird.

Um eine Leerstelle anzuzeigen, verwendet man PRINT " " oder PRINT SPC(1).

Beispiele: A$ = CHR$(65)

Der Zeichenfolge-Variable A$ wird "A" zugewiesen, dessen ASCII Code 65 ist.

PRINT CHR$(107)

Anzeige des graphischen Zeichens " ⊠ ", dessen ASCII Code 107 ist. Zur Festlegung mehrerer ASCII Codenummern hintereinander hält man sich an folgendes Format:

A$ = CHR$(65,66,67,68)

LEN(x$)

x$: Zeichenfolge

Zur Angabe der Anzahl Zeichen der Zeichenfolge x$

Beispiel: A = LEN("ABC")

Der Variablen A wird 3 zugewiesen, dies entspricht der Anzahl Zeichen in der Zeichenfolge "ABC".

(4) Zufallszahl-Funktionen

RND(n)

n: numerischer Wert

Diese Funktion gibt eine Pseudo-Zufallszahl für einen gegebenen numerischen Wert an.

* Eine Pseudo-Zufallszahl zwischen 0.00000001 und 0.99999999 wird generiert.
* Wenn der numerische Wert größer als 0 ist, gibt die Funktion die nächste Pseudo-Zufallszahl im Anschluß an die zuvor gegebene Zahl in der generierten Pseudo-Zufallszahlenserie an.
* Wenn der numerische Wert kleiner oder gleich 0 ist, generiert RND eine neue Pseudo-Zufallszahlenserie, deren erster Wert durch den für X spezifizierten Wert bestimmt wird und gibt die erste Zahl der neuen Serie an. Eine Operation wie z.B. eine Simulation unter Verwendung von Zufallszahlen wird daher periodisch, d.h. kehrt regelmäßig wieder.

Beispiel: Um als Zufallszahl eine Ganzzahl zwischen N und M zu generieren, verwenden wir die folgende Formel:

```
INT (RND(X) * (M - N + 1) + N)
```

Das folgende Programm zeichnet mehrere Kreise. Die Kreisradien und Koordinaten werden durch Zufallszahlen vorgegeben.

```
10 FOR A=1 TO 100
20 B=RND(1)*320
30 C=RND(1)*200
40 D=RND(1)*100
50 E=INT(RND(1)*4)
60 CIRCLE [E,0] B,C,D
70 NEXT A
80 END
```

(5) Joystick-Funktionen

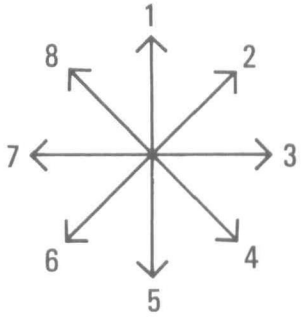
STICK(f)

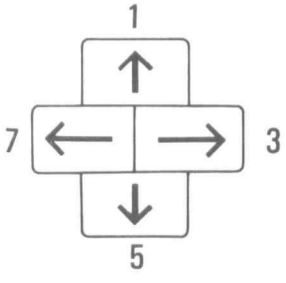
f: numerischer Wert

Angabe einer Ganzzahl von 1 bis 8 zur Anzeige des Joystick Zustands bzw. der Cursorsteuertasten auf der Tastatur. Der numerische Wert f gibt, wie nachstehend gezeigt an, von welchem Gerät die Daten gelesen werden.

0: Cursorsteuertasten der Tastatur
1: Joystick 1
2: Joystick 2

Das Verhältnis zwischen der Ganzzahl und der Richtung, in der der Joystick-Hebel (bzw. die Cursorsteuertasten) gedrückt wird (werden), ist wie folgt.





Cursor-Steuertaste

Wenn durch Vorgabe von 0 als f die Tastatur gewählt wird, so werden, wie nachstehend gezeigt, die Ganzzahlen 2, 4, 6 und 8 angegeben, wenn man zwei Cursorsteuertasten gleichzeitig drückt.

2: [↑] und [→]
4: [→] und [↓]
6: [↓] und [←]
8: [←] und [↑]

STRIG(f)

f: numerischer Wert

Gibt eine Ganzzahl 0 oder 1 an, die den Zustand der Joystick-Taste oder der Leertaste auf der Tastatur anzeigt.

Wenn die Leertaste der Tastatur oder die Joystick-Taste gedrückt wird, wird 1 angegeben. Werden sie nicht gedrückt, dann wird 0 angegeben. Der Ganzzahlwert f bestimmt das Gerät wie folgt:

0: Leertaste der Tastatur
1: Joystick 1-Taste
2: Joystick 2-Taste

Im folgenden Programm kommen die STICK und STRIG Funktionen zur Anwendung. Durch Drücken der Cursortaste wird eine vertikale, horizontale oder schräge Linie gezeichnet und durch Betätigung der Leertaste wird der Bildschirm gelöscht.

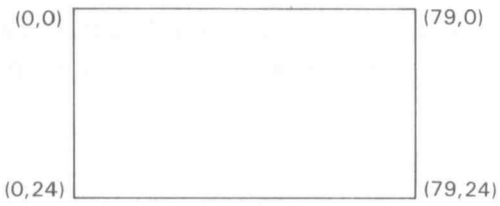
```
10  INIT "CRT:M1"
20  A=STICK(0):B=STRIG(0)
30  ON A GOSUB 200,300,400,500,600,700,800,900
40  IF X<0 THEN X=0
50  IF X>319 THEN X=319
60  IF Y<0 THEN Y=0
70  IF Y>199 THEN Y=199
80  SET X,Y
90  IF B=1 GOTO 10
100 GOTO 20
200 Y=Y-1:RETURN
300 X=X+1:Y=Y-1:RETURN
400 X=X+1:RETURN
500 X=X+1:Y=Y+1:RETURN
600 Y=Y+1:RETURN
700 X=X-1:Y=Y+1:RETURN
800 X=X-1:RETURN
900 X=X-1:Y=Y-1:RETURN
```

5.9 Bildschirmkoordinaten

Bildschirmkoordinaten dienen zur Spezifikation der Bildschirmposition, in der Zeichen und Grafikdaten durch Anzeigebefehle anzuzeigen sind. Diese Koordinaten werden in Form einer horizontalen Position und einer vertikalen Position ausgedrückt. Die Zeichenanzeigepositionen werden unter Verwendung von Zeichenkoordinaten spezifiziert und Zeichenanzeige-Positionen werden unter Verwendung von Grafikkoordinaten spezifiziert.

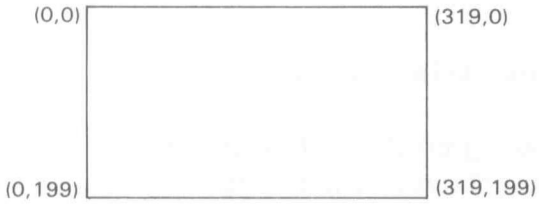
- **Zeichenkoordinaten**

In der 40-Zeichen-Zeile-Betriebsart

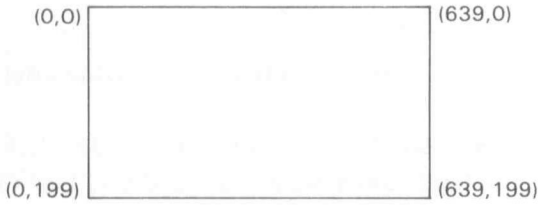


In der 80-Zeichen-Zeile-Betriebsart

- **Grafik-Koordinaten**



320 × 200 Betriebsart



640 × 200 Betriebsart

Die Bereiche beider Zeichenkoordinaten und Grafik-Koordinaten sind je nach Betriebsart verschieden. Die Betriebsart wird mit dem INIT-Befehl spezifiziert.

Kapitel 6 MZ-800 BASIC Befehle und Anweisungen

In diesem Kapitel werden die Befehle und Anweisungen von MZ-800 BASIC erklärt. Die Befehle und Anweisungen sind funktional in die folgenden acht Gruppen aufgeteilt.

— Grundlegende Befehle
— Grundlegende Anweisungen
— Dateisteuer-Anweisungen
— Grafik-Anweisungen
— Musiksteuer-Anweisungen
— Druckersteuer-Anweisungen
— Anweisungen für Maschinensprache-Steuerung
— Fehlerverarbeitungs-Anweisungen

Die Befehle und Anweisungen für die MZ-700 Betriebsart sind in Kapitel 9 zusammengefaßt.

Formatbeschreibungen

Folgende Regeln sind für die Bestimmung von Befehlen, Anweisungen und Funktionen anwendbar.

Winkelklammern "< >" bezeichnen Größen, die vom Anwender bestimmt werden müssen.

Größen in viereckigen Klammern "[ ]" sind zusätzlich.

Größen in { } sind gegenseitig ausschließbar. Nur eine der gezeigten Größen kann bei Anweisungsausführung enthalten sein.

... zeigt an, daß die Größen vor ... wiederholt bestimmt werden können.

6.1 Befehle

AUTO

Format — **AUTO** [<Anfangszeilennummer>][,<Zunahme>]

Abgekürztes Format

A.

Erklärung — Der AUTO-Befehl generiert während des Schreibens eines BASIC-Programms automatisch Zeilennummern. Die Standardvorgabe für beide Parameter ist 10.

Beispiel — (Beispiel 1)

```
AUTO [CR]
10 ........ [CR]
20 ........ [CR]
30 ........ [CR]
```

(Beispiel 2)

```
AUTO 300,5 [CR]
300 ....... [CR]
305 ....... [CR]
310 ....... [CR]
```

Beginnend mit Zeile 300 werden automatisch Programmzeilennummern mit einer Zunahme von je 5 generiert.

(Beispiel 3)

```
AUTO 100 [CR]
100 ....... [CR]
110 ....... [CR]
120 ....... [CR]
```

Beginnend mit Zeile 100 werden Programmzeilennummern mit einer Zunahme von je 10 generiert.

(Beispiel 4)

```
AUTO, 20 [CR]
10 ........ [CR]
30 ........ [CR]
50 ........ [CR]
```

Beginnend mit Zeile 10 werden Programmzeilennummern mit einer Zunahme von je 20 generiert.

Hinweis:
Die Ausführung des AUTO-Befehls wird durch gleichzeitiges Drücken von [SHIFT] und [BREAK] beendet.

DELETE

Format

DELETE [<Anfangszeilennummer> [–] <Endzeilennummer>]
DELETE <Zeilennummer>

Abgekürztes Format

D.

Erklärung

Löscht die Programmzeilen von <Anfangszeilennummer> bis <Endzeilennummer>.

Beispiel

DELETE 150—350 [CR] .... Löscht alle Programmzeilen von 150 bis 350.

DELETE —100 [CR] ..... Löscht alle Programmzeilen bis zu Zeile 100.

DELETE 400— [CR] ..... Löscht alle Programmzeilen von 400 bis zum Programmende.

DELETE 150 [CR] ..... Löscht Zeile 150.

LIST

Format

LIST [/P] [<Anfangszeilennummer>] [–] [<Endzeilennummer>]

Abgekürztes Format

L.

Erklärung

Der LIST-Befehl dient zum Auflisten aller oder eines Teiles der im BASIC-Textbereich enthaltenen Programmzeilen auf dem Bildschirm. Die Ausgabe des Programm-Listings auf dem Bildschirm kann man durch Drücken der Leertaste kurzzeitig unterbrechen; durch nochmaliges Drücken der Leertaste wird das Auflisten fortgesetzt. Um das Auflisten des Programms zu beenden, drückt man die Tasten [SHIFT] + [BREAK].

Durch Eingabe von LIST/P wird das Programm-Listing an den Drucker ausgegeben.

Beispiel

LIST [CR] ..... Zum Auflisten des gesamten Programms.

LIST —30 [CR] ..... Listet alle Zeilen des Programms bis Zeile 30 auf.

LIST 30— [CR] ..... Zum Auflisten aller Programmzeilen ab Zeile 30 bis zum Ende.

LIST 30—50 [CR] ..... Listet alle Programmzeilen ab Zeile 30 bis Zeile 50 auf.

LIST 30 [CR] ..... Listet Zeile 30 des Programms auf.

SEARCH

Format

SEARCH [/P] <Textdaten>

Abgekürztes Format

SE.

Erklärung

Der SEARCH-Befehl sucht das im Speicher stehende BASIC-Programm nach Zeilen ab, die die in <Textdaten> angegebene Zeichenfolge enthalten. Eventuell aufgefundene Zeilen werden auf dem Bildschirm angezeigt. Um als <Textdaten> Anführungszeichen ('') zu spezifizieren, verwendet man CHR$(34). Die Anzeige aufgefundener Zeilen kann durch Drücken der Leertaste unterbrochen werden. Durch nochmaliges Drücken der Leertaste wird die Anzeige fortgesetzt. Um den SEARCH-Befehl zu beenden, drückt man [SHIFT] + [BREAK]. Bei Wahl der /P-Option erfolgt die Ausgabe des SEARCH-Befehls an den Drucker.

Beispiel

SEARCH ''ABC'' ..... Sucht nach Programmzeilen, die die Zeichenfolge "ABC" enthalten und zeigt diese auf dem Bildschirm an.

SEARCH ''PRINT'' + CHR$(34) + ''A'' + CHR$(34) .... Sucht nach Programmzeilen, die den Ausdruck PRINT "A" enthalten.

RENUM

Format

RENUM [<neue Zeilennummer>] [,<alte Zeilennummer>] [,<Zunahme>]

Abgekürztes Format

REN.

Erklärung

Der RENUM-Befehl dient zur Neunumerierung der Zeilen eines BASIC-Programms. Durch die Ausführung dieses Befehls werden die Zeilennummern in Sprungbefehlen wie z.B. GOTO, GOSUB, ON ~ GOTO und ON ~ GOSUB ebenfalls entsprechend neu numeriert.

Beispiel

RENUM ..... Numeriert die Zeilen des zur Zeit im Speicher stehenden Programms neu, so daß die erste Zeilennummer 10 ist, und die Zunahme in Zehnerschritten erfolgt.

RENUM 100 ..... Die Zeilen des zur Zeit im Speicher stehenden Programms werden neu numeriert, so daß sie mit 100 starten und die Zunahme in Zehnerschritten erfolgt.

RENUM 100,50,20 ..... Die Neunumerierung der Zeilen des zur Zeit im Speicher stehenden Programms beginnt mit Zeilennummer 50, die Zeilennummer 50 wird als 100 neu numeriert, woraufhin die Zunahme der nachfolgenden Zeilennummern in Schritten von 20 erfolgt.

(Vor der Neunumerierung)		(Nach der Neunumerierung)
50 A=1		100 A=1
60 A=A+1	⇨	120 A=A+1
70 PRINT A		140 PRINT A
100 GOTO 60		160 GOTO 120

Hinweis:
Wenn neue und alte Zeilennummern spezifiziert werden, muß die angegebene neue Zeilennummer größer sein als die alte Zeilennummer. Bitte beachten, daß ein Fehler verursacht wird, wenn durch Ausführung dieses Befehls eine Zeilennummer größer als 65535 generiert wird.

NEW

Format

NEW

Erklärung

Der NEW-Befehl dient zur Löschung von Programmen im BASIC-Bereich und löscht auch Programmarbeitsbereiche wie den Variablen- und Feldvariablenbereich. Wenn der BASIC-Bereich mit der LIMIT-Anweisung begrenzt ist, löscht der NEW-Befehl nur die Programme im BASIC-Bereich, Maschinenspracheprogramme werden in diesem Falle nicht gelöscht.

Beispiel:

```
10 INPUT A
20 PRINT A
30 END
```

Wenn das obige Programm im Speicher steht, wird es durch Ausführung von NEW gelöscht. (Zur Bestätigung verwendet man den LIST-Befehl.)

NEW ON

Format

NEW ON

Erklärung

Erweitert den BASIC-Programmbereich, indem der Teil des BASIC Interpreters gelöscht wird, der die Ansteuerung des Plotters betrifft. Dieser Befehl kann nur verwendet werden, wenn der optionale Drucker (MZ-1P16) nicht benutzt wird.

Beispiel

NEW ON..... Erweitert den BASIC Programmbereich.

CLR

Format

CLR

Erklärung

Der CLR-Befehl löscht alle Variablen und hebt alle Feldvariablen-Definitionen auf. Alle numerischen Variablen werden gelöscht (werden 0) und den Stringvariablen werden Nullstrings (" ") zugewiesen; die Feldvariablen werden vollständig eliminiert, indem alle zuvor ausgeführten DIM-Anweisungen annulliert werden. Daher müssen in diesem Falle zunächst wiederum DIM-Anweisungen ausgeführt werden, um die Dimensionen der erforderlichen Feldvariablen neu zu definieren, bevor man sie wieder verwenden kann.
Der CLR-Befehl löscht auch alle Funktionsdefinitionen, die mit der DEF FN-Anweisung gemacht wurden; es ist daher auch erforderlich, die DEF FN-Anweisung nochmals auszuführen, um diese Funktion neu zu definieren, bevor man sie wieder verwenden kann.
Der CLR-Befehl darf nicht in FOR ~ NEXT-Schleifen oder Unterprogrammen enthalten sein.

Beispiel

```
10 A=12
20 B$="Monate"
30 PRINT A,B$
40 CLR
50 PRINT A, B$
60 END
```

Die CLR-Anweisung in Zeile 40 löscht die Variablen A und B$, deren Inhalt dadurch jeweils null wird.

CONT

Format **CONT**

Abgekürztes Format

C.

Erklärung

Der CONT-Befehl dient zur Fortsetzung der Ausführung eines Programms, die durch Drücken von [SHIFT] + [BREAK] oder durch eine STOP-Anweisung im Programm unterbrochen wurde. Wenn der Meldung "Ready" ein Punkt (.) nachgestellt ist, kann man den CONT-Befehl verwenden. Beispiele von Situationen, in denen der CONT-Befehl verwendet, bzw. nicht verwendet werden kann, sind in der nachstehenden Tabelle aufgeführt.

Fortsetzung des Programms möglich:	• Die Programmausführung wurde durch Drücken von [SHIFT] + [BREAK] gestoppt. • Die Programmausführung wurde durch eine STOP-Anweisung gestoppt.
Fortsetzung der Programmausführung nicht möglich:	• Vor Ausführung eines RUN-Befehls • Anzeige von "Ready", nachdem während der Programmausführung ein Fehler auftrat. • Der Cassettenbetrieb wurde durch Drücken von [SHIFT] + [BREAK] unterbrochen. • Die Programmausführung wurde während der Ausführung einer MUSIC-Anweisung gestoppt. • Die Programmausführung wurde nach Ausführung einer END-Anweisung gestoppt und "Ready" wird angezeigt.

Siehe hierzu auch

STOP

RUN

Format

RUN [<Anfangszeilennummer>]

Abgekürztes Format

R.

Erklärung

Der RUN-Befehl dient zur Ausführung des zur Zeit im BASIC-Textbereich stehenden Programms. Wenn das Programm ab der ersten Programmzeile ausgeführt werden soll, genügt die Eingabe von RUN und Drücken der [CR]-Taste. Um jedoch das Programm ab einer bestimmten Zeilennummer auszuführen, schreibt man RUN <Anfangszeilennummer> und drückt die [CR]-Taste.
Wenn man diesen Befehl ohne <Anfangszeilennummer> ausführt, löscht der BASIC-Interpreter zunächst alle Variablen und Feldvariablen vor der Ausführung des BASIC-Programms.

Beispiel

RUN ..... Ausführung des Programms vom Anfang an.
RUN 200 ..... Ausführung des Programms ab Zeile 200.

6.2 Grundlegende Anweisungen

CLS

Format — **CLS**

Erklärung — Die CLS-Anweisung dient zur Löschung des gesamten Bildschirms ungeachtet der durch den CONSOLE-Befehl gesetzten Grenzen.

Beispiel — 10 CLS ..... löscht den gesamten Bildschirm.

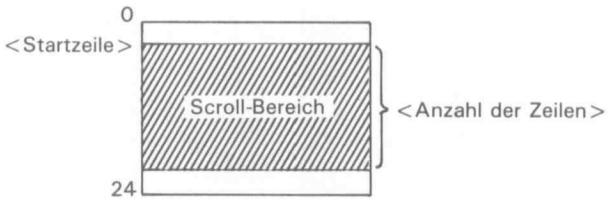
Siehe hierzu auch — CONSOLE

CONSOLE

Format — **CONSOLE** [<Startzeile>, <Anzahl der Zeilen>]

Abgekürztes Format — CONS.

Erklärung — Der CONSOLE-Befehl spezifiziert die Größe des Scroll-Bereichs, d.h des Bereichs, der durch Vorgabe der CLS Anweisung oder Betätigung der Tasten [SHIFT] und [CLR] gelöscht wird. Nach Ausführung eines PLOT ON Befehls ist dieser Befehl ungültig.
Legen Sie beim Editieren einen geeigneten Wert als die <Anzahl der Zeilen> fest, dabei darf die <Anzahl der Zeilen> nicht zu klein sein, da es schwerer ist, innerhalb eines kleinen Scroll-Bereichs Editierarbeiten auf dem Bildschirm vorzunehmen.



Beispiele — CONSOLE 0,25 oder CONSOLE ..... Scrolling (Rollbetrieb des gesamten Bildschirms).
CONSOLE 5,15 ..... Scrolling des Bereichs zwischen den Zeilen 5 und 15 (einschl.).

CURSOR

Format

CURSOR <X-Koordinate>, <Y-Koordinate>

Abgekürztes Format

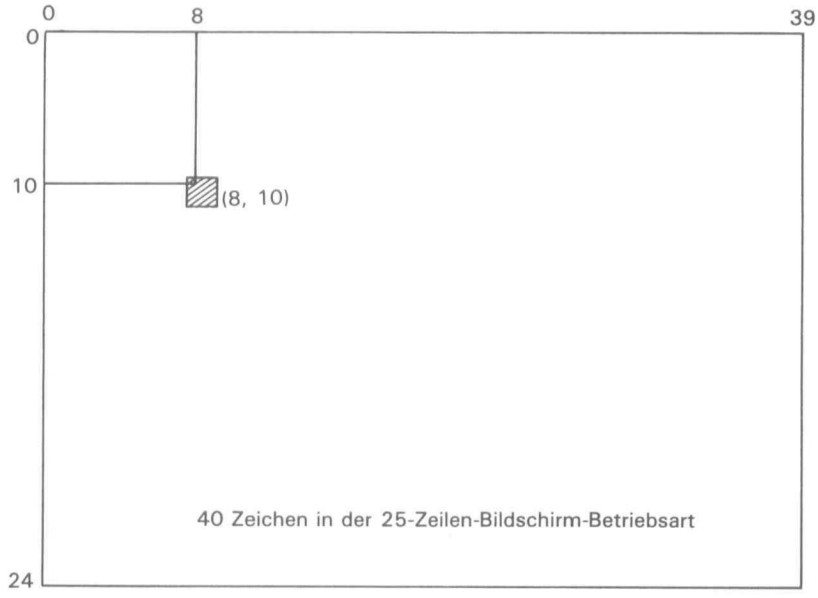
CU.

Erklärung

Die CURSOR-Anweisung bewegt den Cursor zu einer spezifizierten Position auf den Bildschirm. Man kann sie zusammen mit der PRINT und der INPUT-Anweisung zur Anzeige von Zeichen in jeder gewünschten Position verwenden.
Der Wert der <X-Koordinate> muß innerhalb des Bereichs der mit der INIT-Anweisung spezifizierten Bildschirm-Betriebsart liegen. Der Wert der <Y-Koordinate> muß eine Ganzzahl innerhalb von 24 sein. Wenn der für X oder Y spezifizierte Wert keine Ganzzahl ist, so erfolgt vor der Cursorbewegung Umwandlung in eine Ganzzahl, indem die Dezimalstellen abgebrochen werden.

Beispiel

10 CURSOR 8,10 ..... bewegt den Cursor zum Punkt (8,10). Nach Ausführung dieser Anweisung beginnt die Anzeige durch Ausführung einer PRINT oder INPUT-Anweisung ab dieser Stelle.



Siehe hierzu auch

TAB, SPC

REM

Format

REM (remark)

Erklärung

REM ist eine nicht ausführbare Anweisung, die in einer Programmzeile vorgegeben wird, woraufhin der BASIC-Interpreter den Rest der Zeile ignoriert. Da REM-Anweisungen nicht ausgeführt werden, können sie an jeder beliebigen Stelle innerhalb des Programms enthalten sein, ohne daß die Ergebnisse der Ausführung davon betroffen werden. REM-Anweisungen erleichtern im allgemeinen die Lesbarkeit eines Programms oder dienen zur Einfügung von erklärenden Hinweisen.

Beispiele

```
10 REM *** MZ-800 ***
```

LET

Format

LET <Variable> = <Ausdruck>

Erklärung

Die LET-Anweisung weist den durch <Ausdruck> spezifizierten Wert (numerisch oder Zeichenfolge) der durch <Variable> spezifizierten Variable bzw. dem Element einer Feldvariable zu. Wie aus den nachstehenden Beispielen ersichtlich, kann LET auch ausgelassen werden.

Beispiel

```
10 A=10          10 LET A=10
20 PRINT A       20 PRINT A
30 END           30 END
```

Die beiden obigen Programme produzieren genau das gleiche Ergebnis.

```
10 LET N=32
```

Diese Anweisung weist der Variable N den Wert 32 zu.

```
10 LET A=A+5
```

Diese Anweisung addiert 5 zur Variablen A.

```
10 LET B$="SONNTAG"
```

Diese Anweisung weist der Variablen B$ die Zeichenfolge "SONNTAG" zu.

A=1 [CR]

Dies ist ein Beispiel für einen Befehl in der Direktbefehl-Betriebsart.
Der Variablen A wird der Wert 1 zugewiesen.

Nachstehend sind Beispiele unkorrekter Verwendung der LET-Anweisung aufgeführt.

20 LET A$=A+B ..... ungültig, da unterschiedliche Variablentypen (Zeichenfolge und numerisch) auf beiden Seiten des "=" spezifiziert werden.

20 LET LOG (LK)=LK+1 ... Ungültig, da die linke Seite der Anweisung keine numerische Variable bzw. kein Element einer Feldvariable ist.

STOP

Format **STOP**

Abgekürztes Format

S.

Erklärung

Die Programmausführung wird kurzzeitig unterbrochen, die Zeilennummer, bei der die Ausführung stoppt, wird angezeigt, daraufhin wartet das Programm auf die Eingabe eines in der Direktbefehl-Betriebsart ausführbaren Befehls.
Die STOP-Anweisung wird verwendet, um die Programmausführung kurzzeitig zu unterbrechen, man kann sie an allen erforderlichen Stellen im Programm einfügen. Da die Ausführung des Programms nur kurzzeitig unterbrochen wird, kann man die PRINT-Anweisung in der Direktbefehl-Betriebsart verwenden, um z.B. die in den Variablen enthaltenen Werte zu überprüfen, woraufhin die Ausführung durch Eingabe von CONT [CR] fortgesetzt werden kann.

Beispiel

```
10 READ A,B
20 X=A*B
30 STOP
40 Y=A/B
50 PRINT X,Y
60 DATA 15,5
70 END
RUN
Break in 30
Ready.
```

← Dieser Punkt weist darauf hin, daß das Programm durch Ausführung von CONT fortgesetzt werden kann.

Hinweis:
Im Gegensatz zur END-Anweisung werden durch die STOP-Anweisung keine Dateien geschlossen.

Siehe hierzu auch

CONT

END

Format

END

Erklärung

Die END-Anweisung beendet die Programmausführung, woraufhin sich der BASIC-Interpreter wieder in der Direktbefehl-Betriebsart befindet, die die Eingabe von Direktbefehlen (ohne Zeilennummer) ermöglicht. Wenn diese Anweisung ausgeführt wird, erscheint Ready auf der Anzeige, wodurch signalisiert wird, daß der BASIC-Interpreter betriebsbereit ist. Nachdem die END-Anweisung ausgeführt ist, kann die Ausführung auch mit dem CONT-Befehl nicht fortgesetzt werden, selbst wenn sich im Anschluß an die die END-Anweisung enthaltende Programmzeile weitere ausführbare Anweisungen befinden.

Hinweis:
Durch Ausführung der END-Anweisung werden alle offenen Dateien geschlossen.

Unterschiede zwischen der STOP- und der END-Anweisung

	Bildschirmanzeige	Dateien	Fortsetzung der Ausführung
STOP	Break in ×××× Ready.	Offene Dateien werden nicht geschlossen	Fortsetzung durch Ausführung von CONT möglich
END	Ready	Offene Dateien werden geschlossen	Fortsetzung nicht möglich

FOR ~ NEXT

Format

FOR <Laufvariable> = <Ausgangswert> **TO** <Endwert>
[**STEP** <Zunahme>]

NEXT <Laufvariable>

Abgekürztes Format

F. ~ N.

Erklärung

NEXT-Anweisungen dienen dazu, die zwischen FOR und NEXT stehenden Anweisungen eine bestimmte Anzahl Male zu wiederholen.

Beispiel

```
10 A=0
20 FOR N=0 TO 10 STEP 2
30 A=A+1
40 PRINT "N=";N,
50 PRINT "A=";A
60 NEXT N
```

(1) Im obigen Programm wird N der Wert 0 als Ausgangswert zugewiesen.
(2) Daraufhin werden die Zeilen 20 bis 50 ausgeführt und die Werte der Variablen A und N angezeigt.
(3) In Zeile 60 wird der Wert von N um 2 erhöht, woraufhin der BASIC-Interpreter überprüft, ob N größer als 10, d.h größer als der Endwert ist. Andernfalls werden die Zeilen im Anschluß von Zeile 20 wiederholt. Wenn der Wert von N 10 überschreitet, verläßt die Ablaufsteuerung die Schleife und die nachfolgenden Anweisungen (die Zeilen nach Zeile 60) werden ausgeführt. Im obigen Prgramm wird die Schleife sechsmal wiederholt.
Wenn die Angabe STEP (Zunahme) in der Anweisung enthalten ist, wird der Wert der (Laufvariable) bei jedem wiederholten Durchlauf der Schleife um 1 erhöht. Beim obigen Programm würde die Auslassung von STEP 2 zu einer 11-fachen Wiederholung des Schleifenbetriebs führen.

```
FOR  N=0  TO  10  STEP 2
```

Anfangswert von N — Endwert für N — Zunahme für N

FOR ~ NEXT-Schleifen können innerhalb weiterer FOR ~ NEXT-Schleifen verschachtelt werden. Dabei müssen die inneren Schleifen vollständig innerhalb der äußeren enthalten sein. Desgleichen muß für jede Schleife eine separate Laufvariable verwendet werden.

Beispiel

```
10 FOR X=1 TO 9
20 FOR Y=1 TO 9
30 PRINT X*Y;
40 NEXT Y
50 PRINT
60 NEXT X
70 END
```

innere Schleife

äußere Schleife

```
10 FOR A=1 TO 3
20 FOR B=1 TO 5
30 FOR C=1 TO 7
.......
110 NEXT C
120 NEXT B
130 NEXT A
```

110 NEXT C,B,A

Wenn die Schleifen C, B, und A im obigen Beispiel alle am gleichen Punkt enden, genügt es, eine NEXT-Anweisung zu verwenden, um das Ende aller Schleifen anzuzeigen.

Falsches Beispiel:

```
FOR J=1 TO 10
FOR J=K TO K+5
NEXT J
```

Für jede Schleife muß eine eigene Laufvariable verwendet werden.

```
FOR I=1 TO 10
FOR J=K TO K+5
NEXT I
NEXT J
```

Die Schleifen dürfen einander nicht überschneiden.

Hinweis:

Die Anzahl der Verschachtelungen von Schleifen wird durch die BASIC-Syntax nicht begrenzt, zum Speichern von Rücksprungadressen für die einzelnen Ebenen ist jedoch Speicherraum erforderlich, die Anzahl der Verschachtelungen ist daher durch die Größe des verfügbaren freien Speicherraums begrenzt.

Innerhalb einer FOR ~ NEXT-Schleife können die CLR und LIMIT-Anweisungen nicht verwendet werden.

LABEL

Format

LABEL "<Markenname>"

Abgekürztes Format

LA.

Erklärung

Die LABEL-Anweisung definiert eine Marke <Markenname>. Marken werden verwendet, um das Ziel zu definieren, zu dem die Programmausführung von einer GOTO oder GOSUB-Anweisung her übertragen wird. Durch die angemessene Verwendung von Marken im Programm wird die Programmlesbarkeit erheblich verbessert.

Beispiel

```
10 PRINT "BEISPIEL"
20 GOSUB "ABC"
30 PRINT "END"
40 END
100 LABEL "ABC"
110 PRINT "MARKEN-BEISPIEL"
120 RETURN
```

Zeile 100 definiert die Marke "ABC" als Ziel der GOSUB-Anweisung in Zeile 20. Nach Ausführung der GOSUB-Anweisung in Zeile 20 wird die Ablaufsteuerung an das Unterprogramm ab Zeile 100 übertragen.

Siehe hierzu auch

GOTO
GOSUB

GOTO

Format

GOTO {<Zeilennummer> / <Marke>}

Abgekürztes Format

G.

Erklärung

Die GOTO-Anweisung bewirkt eine unbedingte Übertragung der Ablaufsteuerung zu der durch <Zeilennummer> oder <Marke> angegebenen Zeile. Wenn <Zeilennummer> oder <Marke> sich auf eine Zeile bezieht, die ausführbare Anweisungen enthält (d.h andere Anweisungen als REM oder DATA), wird die Ausführung mit dieser Zeile fortgesetzt; andernfalls erfolgt Fortsetzung der Programmausführung ab der ersten ausführbaren Anweisung nach <Zeilennummer> bzw. <Marke>.

Beispiel

```
10 N=1
20 PRINT N
30 N=N+1
40 GOTO 20 ..................... überträgt die Ablaufsteuerung an Zeile 20.
```

Da die Ausführung des obigen Programms endlos fortgesetzt wird, stoppt man es, indem man die [SHIFT] und die [BREAK]-Taste zusammen drückt. (Zum Stoppen der Ausführung eines BASIC-Programms kann dies jederzeit vorgenommen werden). Um die Ausführung fortzusetzen, den CONT-Befehl ausführen.
Die in einer GOTO-Anweisung spezifizierte Zeilennummer darf nicht innerhalb einer FOR ~ NEXT-Schleife stehen.

Siehe hierzu auch

LABEL
GOSUB

ON ~ GOTO

Format

ON <numerischer Ausdruck>
GOTO {<Zeilennummer> [,<Zeilennummer>]...
<Marke> [,<Marke>]...}

Abgekürztes Format

ON ~ G.

Erklärung

Mit der ON ~ GOTO-Anweisung wird die Ausführung je nach dem Wert von <numerischer Ausdruck> an eine der GOTO folgenden Zeilennummern verzweigt. Der Wert von <numerischer Ausdruck> zeigt an, welche der auf GOTO folgenden Zeilennummern für die Verzweigung verwendet werden soll. Mit anderen Worten wird die Ausführung des Programms zur ersten Zeilennummer in der Liste verzweigt, wenn der <numerische Ausdruck> 1 ist, wenn der <numerische Ausdruck> 2 ist, erfolgt Verzweigung zur zweiten Zeilennummer in der Liste usw. Ein Beispiel:

```
100 ON A GOTO 200,300,400,500
```

Zielzeile ist 200 wenn A gleich 1
Zielzeile ist 300 wenn A gleich 2
Zielzeile ist 400 wenn A gleich 3
Zielzeile ist 500 wenn A gleich 4

Beispiel

```
10 INPUT "ZAHL"; A
20 ON A GOTO 50,60,70
30 GOTO 10
50 PRINT "XXX" : GOTO 10
60 PRINT "YYY" : GOTO 10
70 PRINT "ZZZ" : GOTO 10
RUN
ZAHL 1
XXX
ZAHL 2
YYY
ZAHL ■
```

Wenn eine Dezimalzahl wie z.B. 1.2 spezifiziert wird, so werden vor Auswertung der Anweisung die Dezimalstellen abgebrochen.

Hinweis:
Wenn der Wert von <numerischer Ausdruck> in einer ON ~ GOTO-Anweisung größer ist als die im Anschluß an GOTO spezifizierte Anzahl Zeilennummern, wird die Ausführung mit der nächsten Zeile des Programms fortgesetzt.
Dies gilt auch für den Fall, daß der Wert von <numerischer Ausdruck> kleiner als 1 oder negativ ist. Darüberhinaus werden, wenn der Wert von <numerischer Ausdruck> keine Ganzzahl ist, die Dezimalstellen abgebrochen, so daß vor Auswertung der Anweisung ein Ganzzahlwert erhalten wird.

Siehe hierzu auch

GOTO
ON GOSUB

GOSUB ~ RETURN

Format

GOSUB { <Zeilennummer> / <Marke> }

⋮

RETURN

Abgekürztes Format

GOS. ~ RE.

Erklärung

Die GOSUB-Anweisung überträgt die Ablaufsteuerung an ein durch <Marke> gekennzeichnetes Unterprogramm oder an eine durch <Zeilennummer> markierte Zeile. Nachdem die Ausführung des Unterprogramms beendet ist, erfolgt durch die RETURN-Anweisung die Rückkehr zu der auf die GOSUB-Anweisung folgenden Zeile.

Ein Unterprogramm ist eine Folge von Anweisungen, die mehr als einmal innerhalb eines Programms verwendet werden können. Ein Unterprogramm kann auch ein weiteres Unterprogramm aufrufen, das dann seinerseits ein weiteres Unterprogramm aufruft. Die Verschachtelung von Unterprogrammen wird nur durch den verfügbaren Speicherplatz begrenzt. Am Ende des aufgerufenen Unterprogramms muß eine RETURN-Anweisung stehen.

Beispiel

```
10 INPUT A, B
20 GOSUB 100
30 B=C
40 GOSUB 100
50 PRINT C
60 END
```

10 — 60 Hauptprogramm

```
100 C=A↑2+B
110 RETURN
```

100 — 110 Unterprogramm

ON ~ GOSUB

Format

ON <numerischer Ausdruck>
GOSUB {<Zeilennummer>[,<Zeilennummer>]...
<Marke>[,<Marke>]...}

Abgekürztes Format

ON ~ GOS.

Erklärung

Die ON ~ GOSUB Anweisung verzweigt die Programmausführung, je nach dem Wert von <numerischer Ausdruck> zu einer der auf GOSUB folgenden Zeilennummern. Die Arbeitsweise dieser Anweisung ist grundsätzlich die gleiche wie mit der ON ~ GOTO Anweisung, alle Verzweigungen werden jedoch an Unterprogrammen vorgenommen. Nach Rückkehr vom Unterprogramm wird die Ausführung mit der ersten ausführbaren Anweisung fortgesetzt, die auf diejenige ON ~ GOSUB Anweisung folgt, mit der der Unterprogrammaufruf gemacht wurde.

Beispiel

Versuchen wir einmal, die ON ~ GOSUB Anweisung in einem Stundenplan-Programm zu verwenden. Der wichtigste beim folgenden Programm zu beachtende Punkt ist, daß ein Unterprogramm-Aufruf in der Zeile 180 gemacht wird, obwohl die Zeile 180 selbst Teil eines anderen Unterprogramms (von Zeile 170 bis 190) ist, das durch Zeile 90 aufgerufen wird. Unterprogramme können auf diese Weise mehrfach ineinander verschachtelt werden.

```
10  A$=" ENGL " : B$=" MATH " : C$=" FRAN "
20  D$=" PHYS " : E$=" MUSI  " : F$=" SPOR "
30  G$=" GESC " : H$=" KUNS " : I$=" GEOG "
40  J$=" DEUT " : K$=" COMP " : CLS
50  INPUT "Welcher Tag";X$
60  FOR Z=1 TO 7:Y$=MID$("SONMONDIEMITDONFRESAM", 1+3*(Z-1),3):
    IF Y$=X$ THEN X=Z
70  NEXT Z
80  FOR Y=0 TO 4: PRINT TAB(5+6*Y);Y+1;
90  NEXT Y: PRINT
100 ON X GOSUB 180,120,130,140,150,160,170
110 PRINT: GOTO 50
120 PRINT "MO N  ";A$;B$;D$;G$;K$:RETURN
130 PRINT "D I E " ;B$;E$;H$;H$;D$:RETURN
140 PRINT "M I T " ;C$;C$;I$;A$;F$:RETURN
150 PRINT "D O N " ;B$;D$;F$;G$;E$:RETURN
160 PRINT "F R E " ;A$;D$;I$;C$;C$:RETURN
170 PRINT "S A M " ;B$;G$;D$;K$:RETURN
180 FOR Y=1 TO 6
190 ON Y GOSUB 120,130,140,150,160,170
200 PRINT:NEXT Y
210 RETURN
```

IF ~ THEN ~ :ELSE

Format

IF {<Vergleichsausdruck> / <logischer Ausdruck>} **THEN** {<Anweisung> / <Zeilennummer> / <Marke>}

[**:ELSE** {<Anweisung> / <Zeilennummer> / <Marke>}]

Abgekürztes Format

IF ~ TH. ~ :EL.

Erklärung

IF ~ THEN ~ :ELSE Anweisungen werden verwendet, um die Ausführung des Programms in Abhängigkeit vom Ergebnis eines logischen Ausdrucks oder eines Vergleichsausdrucks zu verzweigen. Wenn das Ergebnis eines derartigen Ausdrucks wahr ist, wird die auf THEN folgende Anweisung ausgeführt. Wenn THEN eine Zeilennummer nachgestellt ist, erfolgt Sprung der Programmausführung zu dieser Zeile, vorausgesetzt, daß das Ergebnis des vorherigen Ausdrucks wahr ist.
Wenn das Ergebnis des logischen Ausdrucks bzw. des Vergleichsausdrucks falsch ist, wird (werden) die auf ELSE folgende(n) Anweisung(en) ausgeführt. Wenn nach ELSE eine Zeilenummer spezifiziert ist, erfolgt ein Sprung in der Programmausführung zu dieser Zeile. Wird hingegen :ELSE ausgelassen und das Ergebnis des Ausdrucks ist falsch, so wird die Ausführung mit der Zeile fortgesetzt, die auf die die IF ~ THEN Anweisung enthaltende Zeile folgt.

(Wenn ELSE nicht verwendet wird.)

(Wenn ELSE nicht verwendet wird.)

Beispiel

```
10 IF C <1 THEN C=3 :ELSE C=C-1
```

Diese Anweisung weist der Variablen C den Wert 3 zu, wenn C kleiner als 1 ist; in allen anderen Fällen wird C der Wert C – 1 zugewiesen.

```
10 IF C< > D THEN 150 :ELSE END
```

Diese Anweisung bewirkt Sprung zur Zeile 150, wenn C nicht gleich D ist; anderenfalls wird die Programmausführung beendet.

```
10 IF A$="ABC" THEN A$=A$+"DEF"
```

Diese Anweisung weist A$ die Zeichenfolge "ABCDEF" zu, vorausgesetzt, daß A$ die Zeichenfolge "ABC" enthält, anderenfalls erfolgt einfach nur Fortsetzung mit der nächsten Programmzeile.

Hinweis (Vorsichtsmaßnahmen zum Vergleich von numerischen Werten mit BASIC 1Z-016):
Numerische Werte werden intern in Binärkomma-Schreibweise (Gleitkomma) dargestellt. Da diese Werte zur Weiterverarbeitung oder externen Anzeige (z.B. mit der PRINT-Anweisung im Dezimalformat) umgewandelt werden müssen, tritt möglicherweise ein durch die Umwandlung bedingter Fehler im Betrag auf.

Wenn z.B. ein arithmetischer Ausdruck ausgewertet wird, dessen Rechenergebnis eine Ganzzahl ist, wird möglicherweise nach Ausführung der Operation kein Ganzzahlwert angegeben, wenn andere Werte als Ganzzahlen während der Ausführung der Rechnungen verarbeitet werden. Dies ist besonders bei der Auswertung von Vergleichsausdrücken unter Verwendung von "=" zu berücksichtigen.
Wir veranschaulichen dies mit dem nachstehenden Beispielprogramm, in dem das Ergebnis FALSCH angezeigt wird, nachdem das Programm geprüft hat, ob 1 und 1/100*100 gleich sind.

```
10 A=1/100*100
20 IF A=1 THEN PRINT "RICHTIG":ELSE PRINT"FALSCH"
30 PRINT "A=";A
40 END
RUN
FALSCH
A= 1
```

Die Tatsache, daß als Ergebnis der Ausführung dieses Programms sowohl "FALSCH" als auch "A = 1" angezeigt werden, zeigt, daß die externe Darstellung einer Zahl sich gegebenenfalls von der internen Darstellung der Zahl unterscheiden kann.

IF ~ GOTO

Format

IF {<Vergleichsausdruck> / <logischer Ausdruck>} **GOTO** {<Zeilennummer> / <Marke>}

Abgekürztes Format

IF ~ G.

Erklärung:

Die IF ~ GOTO Anweisungsfolge wertet die durch <Vergleichausdruck bzw. logischer Ausdruck> vorgegebene Bedingung aus und verzweigt dann die Programmausführung zu der durch <Zeilennummer> oder <Marke> spezifizierten Zeile, wenn die Bedingung erfüllt ist. Wie bei der IF THEN-Folge wird auch IF ~ GOTO für bedingtes Verzweigen verwendet, d.h es erfolgt nur dann ein Sprung der Programmausführung zu der in <Zeilennummer> oder <Marke> angegebenen Zeile, wenn die vorgegebene Bedingung erfüllt wird. Wenn die Bedingung nicht erfüllt wird, so wird die Programmausführung mit der nächsten Programmzeile fortgesetzt. (Alle auf IF ~ GOTO in der gleichen Programmzeile folgenden Anweisungen werden ignoriert.

Beispiel

```
10 T=0:N=0
20  INPUT ''Wert='';X
30  IF X=999 GOTO 100
40  T=T+X:N=N+1
50  GOTO 20
100 PRINT ''*********************''
110 PRINT ''SUMME:'';T
120 PRINT ''Anzahl der Eingaben:'';N
130 PRINT ''Durchschnitt:'';T/N
140 END
```

Im obigen Beispiel wird die Summe und der Durchschnitt der eingegebenen Werte ermittelt. Wenn man 999 eingibt, wird die Programmausführung beendet.

Siehe hierzu auch

GOTO
IF ~ THEN ~ :ELSE
IF ~ GOSUB

IF ~ GOSUB

Format

IF {<Vergleichsausdruck> / <logischer Ausdruck>} **GOSUB** {<Zeilennummer> / <Marke>}

Abgekürztes Format

IF ~ GOS.

Erklärung

Die IF GOSUB-Anweisung wertet die durch <Vergleichsausdruck> oder <logischer Ausdruck> vorgegebene Bedingung aus und verzweigt den Programmablauf zu dem Unterprogramm, das mit der durch <Zeilennummer> oder <Marke> spezifizierten Zeile beginnt. Nach Ausführung des Unterprogramms erfolgt Rückkehr der Ablaufsteuerung zur ersten ausführbaren Anweisung im Anschluß an die für den Unterprogrammaufruf verwendete IF ~ GOSUB-Anweisung; wenn daher die Zeile mit der IF ~ GOSUB-Anweisung mehrere weitere Anweisungen enthält, wird die Programmausführung ab der ersten auf IF ~ GOSUB folgenden Anweisung fortgesetzt.

Beispiel

```
10 INPUT "X=";X
20 IF X<0 GOSUB 100:PRINT "X<0"
30 IF X=0 GOSUB 200:PRINT "X=0"
40 IF X>0 GOSUB 300:PRINT "X>0"
50 PRINT "*********************"
60 GOTO 10
100 PRINT "* Programmzeile 100": RETURN
200 PRINT "* Programmzeile 200": RETURN
300 PRINT "* Programmzeile 300": RETURN
```

Siehe hierzu auch

GOSUB ~ RETURN
IF ~ THEN ~ :ELSE
IF ~ GOTO

PRINT

Format

PRINT [<Palettencode>] <Daten> [{;, } <Daten>]...

Abgekürztes Format

?

Erklärung

Die PRINT-Anweisung dient zur Anzeige von Daten auf dem Bildschirm. <Palettencode> spezifiziert den Palettencode zur Wahl der Farbe des angezeigten Textes. Wenn man diesen Code nicht angibt, wird der in der COLOR-Anweisung spezifizierte Palettencode vorausgesetzt.
Ein zur Begrenzung von zwei <Daten>-Posten verwendetes Semikolon bewirkt, daß diese ohne Zwischenraum hintereinander angezeigt werden. Verwendet man andererseits ein Komma, werden die beiden <Daten>-Posten mit einem eingefügten zehn Zeichen langen Tabulatorabstand auf dem Bildschirm angezeigt.
Numerische Daten werden durch diese Anweisung in einem der folgenden zwei Formate angezeigt: Als reelle Zahl oder im Exponentialformat. Numerische Werte im Bereich von 1×10^{-8} bis 1×10^{8} werden als reelle Zahl angezeigt, über diesen Bereich hinausgehende Zahlen werden im Exponentialformat dargestellt.

Beispiel

10 PRINT [2] "ABC";123..... Die Textdaten "ABC" und die numerischen Daten 123 werden gemäß dem vorgegebenen Palettencode 2 ohne eingefügten Abstand, d.h in einer Reihe hintereinander auf dem Bildschirm angezeigt.

20 PRINT [2] "ABC",123..... Die Textdaten "ABC" und die numerischen Daten 123 werden durch einen eingefügten 10 Zeichen langen Tabulatorabstand voneinander getrennt angezeigt.

Nachstehend sind einige Spezialverwendungen der PRINT-Anweisung aufgeführt.

PRINT " C " Löscht den gesamten Bildschirm und bewegt den Cursor in Ausgangsposition (obere linke Ecke) auf dem Bildschirm.
PRINT " H " Ohne Löschen des Bildschirms wird der Cursor in die Ausgangsposition gebracht.
PRINT " → " bewegt den Cursor um eine Stelle nach rechts.
PRINT " ← " bewegt den Cursor um eine Stelle nach links.
PRINT " ↑ " bewegt den Cursor um eine Zeile aufwärts.
PRINT " ↓ " bewegt den Cursor um eine Zeile abwärts.

Zur Eingabe von Spezialzeichen für die Cursorsteuerung betätigt man zunächst die [GRAPH]-Taste; woraufhin der Cursor folgende Form annimmt: "_". Als nächstes drückt man eine der Editiertasten [CLR], [HOME], [→], [←], [↑], oder [↓]. Nach der Eingabe des Spezialzeichens kehrt man durch Drücken der [ALPHA]-Taste dann wieder in die ursprüngliche Betriebsart zurück.

Siehe hierzu auch

COLOR, PAL

PRINT USING

Format

PRINT [<Palettencode>]
USING <Format-Zeichenfolge>; <Daten> [{ ; , }<Daten>]...

Abgekürztes Format

? USI.

Erklärung

Die PRINT USING-Anweisung dient zur Anzeige der Daten auf dem Bildschirm in einem ganz bestimmten Format. Abgesehen von der Angabe der <Format-Zeichenfolge> wird diese Anweisung im gleichen Format spezifiziert wie die PRINT-Anweisung. Die <Format-Zeichenfolge> besteht aus Formatierzeichen. Diese Formatierzeichen legen, wie in den nachstehenden Beispielen beschrieben, das Format der anzuzeigenden Daten fest.

(1) Formatierzeichen für numerische Werte

(a) #

Zur Darstellung jeder einzelnen Stellenposition wird ein Zahlenzeichen verwendet. Wenn die anzuzeigende Zahl weniger Stellen hat als Positionen vorgegeben sind, erfolgt entsprechender Rechtsausgleich der Zahl.

```
10 A=123
20 PRINT USING "# # # #";A
RUN
␣123
```

(b) .

Ein Punktzeichen markiert die Position, in der die Kommasetzung erfolgen soll. Die Anzahl der Zeichen rechts vom Dezimalpunkt schreibt die Anzahl der anzuzeigenden Dezimalstellen vor.

```
10 A=12
20 PRINT USING "# # #.# #";A
RUN
␣12.00
```

(c) ,

Durch Einfügen eines Kommas nach jedem dritten Zeichen in der <Format-Zeichenfolge> wird eine übersichtliche Tausenderunterteilung vorgenommen. Die Anzeige der Zahlen erfolgt mit Rechtsausgleich.

```
10 A=6345123
20 PRINT USING "#,# # #,# # #";A
RUN
6,345,123
```

(d) + und –

Um das Vorzeichen der anzuzeigenden Zahl zu bestimmen, kann ein Pluszeichen (+) oder ein Minuszeichen (–) am Ende der (Format-Zeichenfolge) enthalten sein, das gegebenenfalls in eine Leerstelle abgewandelt wird. Zum Beispiel bewirkt die Vorgabe PRINT USING "# # # # +"; A oder PRINT USING "# # # # –"; A die Anzeige des Vorzeichens unmittelbar im Anschluß an die Zahl. (PRINT USING "# # # # –") bewirkt die Anzeige eines Minuszeichens im Anschluß an die Zahl, vorausgesetzt, daß die Zahl negativ ist; falls die Zahl positiv ist, bleibt die Anzeige dieser Position leer). Außerdem kann am Anfang der Format-Zeichenfolge ein Pluszeichen spezifiziert werden, um zu bestimmen, daß ungeachtet der Tatsache, ob es sich um eine positive oder eine negative Zahl handelt, das Vorzeichen der Zahl angezeigt werden soll.

```
PRINT USING "# # # # +";-13
␣␣13-
PRINT USING "+ # # # #";25
␣␣+25
```

Hinweis:

Obwohl auch ein Minuszeichen am Anfang der Format-Zeichenfolge spezifiziert werden kann und auch angezeigt wird, steht dieses in keiner Beziehung zu dem tatsächlichen Vorzeichen der Zahl.

(e) * *

Wenn man am Anfang der Format-Zeichenfolge zwei Sternsymbole spezifiziert, weist dies darauf hin, daß in den Positionen der Vorlaufnullen Sternsymbole anzuzeigen sind.

```
10 A=123
20 PRINT USING "* * # # # #";A
RUN
* * *123
```

(f) ££

Die Vorgabe von zwei Pfundzeichen am Anfang der Format-Zeichenfolge weist darauf hin, daß ein Pfundzeichen in der Position unmittelbar links von der Zahl anzuzeigen ist

```
10 A=123
20 PRINT USING "££# # # #";A
RUN
␣␣£123
```

(g) $$

Die Vorgabe von zwei Dollarzeichen am Anfang der Format-Zeichenfolge weist darauf hin, daß unmittelbar links von der Zahl ein Dollarzeichen anzuzeigen ist.

```
10 A=456
20 PRINT USING "$$# # # #";A
RUN
␣␣$456
```

(h) ↑↑↑↑

Wenn am Ende der Format-Zeichenfolge vier Exponenten-Operatorzeichen enthalten sind, wird angewiesen, daß die Zahl im Exponentialformat angezeigt werden soll.

```
10 A=51123
20 PRINT USING "# #.# # #↑↑↑↑";A
RUN
␣5.112E+04
```

In diesem Falle ist die erste Position zur Anzeige des Vorzeichens der Zahl reserviert.

(i) Erweiterte Operandenliste

Eine Variablenliste kann im Anschluß an eine einzelne PRINT USING-Anweisung spezifiziert werden, wobei die einzelnen Variablen durch Kommas oder Semikolon voneinander getrennt werden. Mit dieser Vorgabe wird das in der (Format-Zeichenfolge) spezifizierte Format zur Anzeige aller resultierenden Werte verwendet.

```
10 A=5.3: B=6.9: C=7.123
20 PRINT USING "# #.# #"; {A;B;C}
                          {A,B,C}
RUN
␣5.300␣6.900␣7.123
```

Ob man nun Semikolons oder Kommas zur Trennung der Variablen verwendet, hat keinen Einfluß auf das Ergebnis.

(2) Formatierzeichen für Zeichenfolgen (Strings)

(a) !

Ein Ausrufezeichen in der <Format-Zeichenfolge> schreibt vor, daß nur das erste Zeichen in der gegebenen Zeichenfolge angezeigt werden soll.

```
10 A$="CDE"
20 PRINT USING "!";A$
RUN
C
```

(b) &␣␣␣␣&

Zwei &-Zeichen mit n Leerstellen dazwischen schreiben vor, daß die ersten 2+n Zeichen in der spezifizierten Zeichenfolge angezeigt werden sollen. Wenn die Zeichenfolge kürzer ist als das durch <&␣␣␣␣&> definierte Feld, erfolgt Linksausgleich und Auffüllen der Zeichenfolge mit Leerzeichen nach rechts. Wenn die Zeichenfolge länger ist als das Feld werden die darüber hinausgehenden Zeichen ignoriert.

```
10 A$="ABCDEFGH"
20 PRINT USING "&␣␣␣␣&";A$
RUN
ABCDEF

10 A$="XY"
20 PRINT USING "&␣␣␣&";A$
RUN
XY␣␣␣
```

(3) Stringkonstanten-Ausgabefunktion

Wenn andere Zeichen als die oben genannten in der Format-Zeichenfolge einer PRINT USING-Anweisung enthalten sind, werden diese Zeichen zusammen mit dem auf das Semikolon folgenden Wert angezeigt.

```
10 A=123
20 PRINT USING "DATEN ####";A
RUN
DATEN ␣123
```

(4) Abtrennen der USING-Klausel

Normalerweise werden die Begriffe PRINT und USING unmittelbar aufeinanderfolgend verwendet; man kann sie jedoch innerhalb der gleichen Anweisung auch getrennt verwenden.

```
10 A=-12 : B=14 : C=12
20 PRINT A;B USING "####";C
RUN
-12␣14␣␣12
```

Im obigen Beispiel enthält Zeile 20 eine normale PRINT-Anweisung und eine USING-Klausel.

INPUT

Format **INPUT** [<Meldung>;]<Variable>[,<Variable>]...

Abgekürztes Format

I.

Erklärung Die INPUT-Anweisung liest während der Programmausführung eingegebene Daten und weist sie der <Variable> zu.
Wenn während der Programmausführung eine INPUT-Anweisung auftritt, wird der Ablauf an dieser Stelle unterbrochen, ein Fragezeichen erscheint auf dem Bildschirm und der CURSOR blinkt rechts daneben, um anzuzeigen, daß das Programm auf die Eingabe von Daten wartet. Wenn eine <Meldung> spezifiziert ist, wird der Text dieser Meldung anstelle des Fragezeichens angezeigt. Nach Eingabe der Daten über die Tastatur und Drücken der [CR]-Taste, wird der eingegebene Wert der <Variable> zugewiesen und die Programmausführung fortgesetzt. Die Daten und <Variable> müssen hinsichtlich Typ übereinstimmen. Zeichenfolgen können ohne Anführungsstriche eingegeben werden. Dabei werden evtl. mit eingegebene vor- oder nachgestellte Leerstellen ignoriert. Wenn eingegebene vor- oder nachgestellte Leerstellen oder Kommas in der Zeichenfolge enthalten sein sollen, müssen Anfang und Ende der eingegebenen Zeichenfolge von Anführungsstrichen begrenzt werden.

Beispiel

10 INPUT A,B$ ..... Gestattet die Eingabe von Daten und zeigt Fragezeichen an.
Nachdem die Dateneingabe beendet ist, werden die ersten Daten der Variable A und die zweiten Daten der Stringvariable B$ zugewiesen.

20 INPUT "A="; A..... Zeigt die Meldung "A=" an und wartet auf eine Dateneingabe.

GET

Format

GET <Variable>

Erklärung

Die GET-Anweisung prüft, ob eine Taste der Tastatur gedrückt wurde und, falls, dem so ist, wird der durch <Variable> spezifizierten Variablen der entsprechende Wert zugewiesen. Wenn keine Taste betätigt wurde, bleibt die Variable leer (0 im Falle von numerischen Variablen bzw. null für eine Stringvariable).
Bei numerischen Variablen gestattet diese Anweisung die Eingabe einer einzigen Ziffer (von 0 bis 9). Bei Stringvariablen ist die Eingabe eines einzigen Zeichens gestattet. Eingaben nichtnumerischer Werte für eine numerische Variable werden ignoriert.

Beispiel

```
10 GET A$: IF A$=" " THEN 10
20 PRINT A$
30 END
```

Dieses Programm bewirkt die Anzeige eines über die Tastatur eingegebenen Zeichens, vorausgesetzt, daß es sich um ein ein druckbares Zeichen handelt.

DIM

Format

DIM <Variable> (<Index>)[,<Variable>(<Index>)]...
DIM <Variable>(<Index>,<Index>)[,<Variable>(<Index>,<Index>)]...

Erklärung

Durch die DIM-Anweisung werden Feldvariablen mit einer bis vier Dimensionen festgelegt und der Speicherplatz für die Anzahl der Elemente dieser Feldvariablen reserviert (DIM: Dimension).
Max. zwei alphanumerische Zeichen, von denen der erste ein Großbuchstabe sein muß, können als Feldvariablenname für <Variable> bestimmt werden. Indices dienen zur Bestimmung der Anzahl der Dimensionen (durch die Anzahl der Indices) und der Anzahl Elemente (durch den Indexwert), wobei jedoch in der Praxis durch die Größe des verfügbaren Speicherraums effektiv Grenzen gesetzt werden. Für jede vereinbarte Feldvariable muß es einen eigenen, unverkennbaren Namen geben, so ist z.B. die Festlegung DIM A(5), A(6,3) nicht zulässig.
Durch Ausführung einer DIM-Anweisung werden die Werte aller Elemente der festgelegten Feldvariablen auf 0 (im Falle von numerischen Feldvariablen) bzw. auf null (im Falle von String-Feldvariablen) gesetzt. Die Ausführung dieser Anweisung sollte daher bereits abgeschlossen sein, bevor den Feldvariablen Werte zugewiesen werden.
Wenn die DIM-Anweisung für eine bereits festgelegte Feldvariable nochmals ausgeführt wird und die neue Anweisung sieht eine größere Dimensionierung vor als die alte, tritt ein Fehler auf.
Durch Ausführung eines CLR-Befehls bzw. eines NEW-Befehls werden alle Festlegungen von Feldvariablen anulliert.

Beispiele

10 DIM A(3) ..... Legt eine eindimensionale numerische Feldvariable A mit 4 Elementen fest.

A(0)	A(1)	A(2)	A(3)

$3+1=4$ Elemente

20 DIM B$(2,3)..... Legt eine zweidemensionale Stringfeldvariable B mit 12 Elementen fest.

B$(0,0)	B$(0,1)	B$(0,2)
B$(1,0)	B$(1,1)	B$(1,2)
B$(2,0)	B$(2,1)	B$(2,2)
B$(3,0)	B$(3,1)	B$(3,2)

$(2+1)\times(3+1)=12$ Elemente

```
10 DIM A(2)
20 FOR J=0 TO 2
30 INPUT A(J)
40 NEXT J
50 PRINT A(0), A(1), A(2)
60 END
```

Drei Feldvariablen (A(0), A(1), und A(2)) werden in diesem Beispiel verwendet. Das Programm gibt in diese Variablen drei Zahlen ein und zeigt daraufhin diese Zahlen an.

READ ~ DATA

Format

READ <Variable>[, <Variable>]...

⋮

DATA <Konstante>[, <Konstante>]...

Abgekürztes Format

REA. ~ DA.

Erklärung

Wie die INPUT und GET-Anweisung wird auch die READ-Anweisung dafür verwendet, dem Computer Daten zur Verarbeitung zukommen zu lassen. Im Gegensatz zur INPUT und GET-Anweisung werden die Daten jedoch nicht über die Tastatur eingegeben, sondern in DATA-Anweisungen im Programm selbst gespeichert. Genauer gesagt besteht die Funktion der READ-Anweisung darin, die in einer DATA-Anweisung enthaltene Liste von Werten bzw. Zeichenfolgen nacheinander in Variablen einzulesen. Dabei muß die Anzahl der Variablen der READ-Anweisungen und die Anzahl der in den DATA-Anweisungen enthaltenen Daten im Verhältnis 1:1 übereinstimmen. Bei in DATA-Anweisungen enthaltenen Zeichenfolgen können Anführungsstriche ausgelassen werden. Bei Nullstrings dürfen sie jedoch nicht ausgelassen werden.

Beispiel

(Beispiel 1)

```
10 READ A,B,C,D
20 PRINT A;B;C;D
30 END
40 DATA 10,100,50,60
RUN
 10 100 50 60
```

In diesem Beispiel werden die in der DATA-Anweisung spezifizierten Werte durch die READ-Anweisung in die Variablen A,B,C und D eingelesen, woraufhin die Werte dieser Variablen angezeigt werden.

(Beispiel 2)

```
10 READ X$,A1,Z$
20 PRINT X$;A1;Z$
30 END
40 DATA A,1,C
```

Wie aus diesem Beispiel ersichtlich, müssen die in DATA-Anweisungen enthaltenen Zeichenfolge-Daten nicht von Anführungszeichen eingeschlossen sein.

```
RUN
A 1C
```

Die READ-Anweisung in diesem Beispiel greift nacheinander auf die Liste der in der DATA-Anweisung spezifizierten Daten zu, und weist die einzelnen Daten den entsprechenden, im Anschluß an die READ-Anweisung aufgeführten Variablen zu.

(Beispiel 3)

```
10 DIM (2)
20 READ A(0),A(1),A(2)
30 PRINT A(0);A(1);A(2)
40 END
50 DATA 3,4,5
RUN
 3 4 5
```

Die READ-Anweisung in diesem Programm weist die in der DATA-Anweisung enthaltenen numerischen Werte den Feldvariablen A(0), A(1) und A(2) zu, daraufhin werden die Werte dieser Feldvariablen-Elemente mit der PRINT-Anweisung in Zeile 30 auf dem Bildschirm angezeigt.

(Beispiel 4)

```
10 READ A
20 READ B
30 DATA X
```

Das obige Beispiel ist falsch, da erstens durch die READ-Anweisung in Zeile 10 eine numerische Variable spezifiziert wird , aber der im Anschluß an die DATA-Anweisung spezifizierte Wert eine Zeichenfolge ist und zweitens, da es keine Daten gibt, die durch die READ-Anweisung in Zeile 20 gelesen werden können.

Siehe hierzu auch

RESTORE

RESTORE

Format

RESTORE [{<Zeilennummer> / <Marke>}]

Abgekürztes Format

RES.

Erklärung

Wenn die RESTORE-Anweisung ohne Zeilennummer oder mit der Zeilennummer 0 spezifiziert wird, so wird bewirkt, daß der BASIC-Interpreter (bei Auftreten von READ-Anweisungen) die Liste der Daten noch einmal von Anfang an liest, d.h ab der DATA-Anweisung mit der kleinsten Zeilennummer. Wenn <Zeilennummer> oder <Marke> spezifiziert wird, so wird durch Ausführung dieser Anweisung bewirkt, daß der BASIC-Interpreter das Lesen der Daten ab der durch <Zeilennummer> oder <Marke> spezifizierten Zeile liest, bzw. falls diese Zeile keine DATA-Anweisung enthält, ab der nächsten DATA-Anweisung im Anschluß an diese Zeile.

Beispiel

```
10  DATA ''PERSONAL COMPUTER''
20  DATA ''MZ-800
30  READ A$,B$
40  PRINT A$;B$
50  RESTORE 20
60  READ C$
70  PRINT C$
80  RESTORE
90  READ D$
100 PRINT D$
110 END
RUN
PERSONAL COMPUTER MZ-800''
MZ-800
PERSONAL COMPUTER
```

Siehe hierzu auch

READ ~ DATA

DEF FN

Format

DEF FN <Funktionname> (<Variable>) = <numerischer Ausdruck>

Erklärung

Die DEF FN-Anweisung wird verwendet, um die Benutzerfunktion FN f(x) zu definieren. Derartige Funktionen bestehen aus Kombinationen von BASIC zu eigenen Funktionen. Der <Funktionsname> ist ein Großbuchstabe.

Beispiel

DEF FNA (X) = 2*X↑2+3*X+1 ... Definiert $2X^2+3X+1$ als FNA (X).
DEF FNE(V) = 1/2*M*V↑2..... Definiert $1/2MV^2$ als FNE (V)

(Falsche Definitionen)

```
10 DEF FNK(X)=SIN(X/3+π/4), FNL(X)=EXP(-X↑2/K)
```
..... Mit einer einzigen DEF FN-Anweisung kann nur eine Benutzerfunktion definiert werden.

```
10 DEF FND(X)=FNB(X)/C+X
```
..... Mit DEF FN definierte Funktionen können nicht in anderen DEF FN Anweisungen verwendet werden.

Zu ermitteln ist die kinetische Energie einer Masse von 5,5 kg wenn sie Anfangsbeschleunigungen von 3.5 m/s^2, 3.5 ×2 m/s^2 und 3.5 ×3 m/s^2 ausgesetzt wird.

```
10 DEF FNE(V)=1/2*M*V↑2
20 M=5.5:V=3.5
30 PRINT FNE(V), FNE(V*2), FNE(V*3)
40 END
```

Hinweis:
Durch Ausführung des CLR oder NEW-Befehls werden alle definierten Benutzerfunktionen gelöscht.

TRON

Format

TRON [/P]

Abgekürztes Format

TR.

Erklärung

Der TRON-Befehl dient zur Verfolgung der Programmausführung. Die Ausführung eines TRON-Befehls bewirkt die Anzeige der Zeilennummern, in Klammern eingeschlossen, auf dem Bildschirm gemäß der Abarbeitung des Programms durch den BASIC-Interpreter. Mit der /P-Option erfolgt die Ausgabe der Zeilennummern mit dem TRON-Befehl an den Drucker.

Beispiel

```
10  DEF FNA(X,Y)=X*Y
20  READ A1,A2,A3,A4
30  W=FNA(A1,A2);GOSUB 100
40  W=FNA(A2,A3):GOSUB 100
50  W=FNA(A3,A4):GOSUB 100
60  DATA 4,5,6,7
70  END
100 IF W>20 THEN PRINT "ABCD"
110 RETURN
```

Geben Sie vor Ausführung des Programms den Direktbefehl TRON ein.

```
RUN
[10][20][30][100][110][40][100]ABCD
[110][50][100]ABCD
[110][60][70]
```

Die Zeilennummern erscheinen jeweils dann auf dem Bildschirm, wenn ihre Ausführung erfolgt; man kann daher den Programmablauf am Bildschirm genau verfolgen. Um die Verfolgungs-Betriebsart zu beenden, gibt man den TROFF-Befehl ein.

Siehe hierzu auch

TROFF

TROFF

Format **TROFF**

Abgekürztes Format

TROF.

Erklärung Der TROFF-Befehl setzt die Verfolgungs-Funktion außer Betrieb.

Siehe hierzu auch

TRON

DEF KEY

Format **DEF KEY** (<Tastennummer>) = "<Zeichenfolge>"

Erklärung Die zehn Funktionstasten können vom Benutzer mit beliebigen Zeichenfolgen belegt werden, so daß die gewünschte Zeichenfolge jederzeit mit einem Tastendruck eingegeben werden kann. Dieser Befehl dient dazu, diese Zeichenfolgen zu definieren und sie den vom Benutzer definierbaren Funktionstasten zuzuweisen.
Zur Verwendung der Funktionstasten 1 bis 5 genügt es, die entsprechende Taste, die sich im oberen linken Teil der Tastatur befindet, zu drücken; zur Verwendung der Tasten 6 bis 10 betätigt man die zugehörige Funktionstaste bei gedrückter [SHIFT]-Taste. Die Nummer der Funktionstaste (1 bis 10) wird durch <Tastennummer> spezifiziert und die der Taste zuzuweisende Zeichenfolge bzw. die gewünschte Anweisung wird so wie sie ist in der <Zeichenfolge> vorgegeben. Die <Zeichenfolge> darf einschließlich der Leerstellen eine Länge von 15 Zeichen nicht überschreiten. Durch Ausführung der DEF KEY-Anweisung wird die bisherige Festlegung für die definierbare Funktionstaste aufgehoben.

Beispiel

```
10 DEF KEY (1)="SHARP"................ Für die Funktionstaste [F1] wird die
                                        Zeichenfolge SHARP vorgegeben.
20 DEF KEY (2)="RUN"+CHR$ (13)... Für die Funktionstaste [F2] wird die
                                        Zeichenfolge RUN [CR] vorgegeben.
```

Hinweis:
CHR$(13) entspricht dem ASCII-Code für CR (Wagenrücklauf). Wenn man der definierbaren Funktionstaste diesen Code zusammen mit der Zeichenfolge zuweist, hat dies die gleiche Wirkung wie eine manuelle Betätigung der [CR]-Taste.

KEY LIST

Format **KEY LIST**

Abgekürztes Format

K.L.

Erklärung Mit dem KEY LIST-Befehl werden die den definierbaren Funktionstasten zugewiesenen Zeichenfolgen der Reihe nach angezeigt.

Beispiel

```
KEY LIST [CR]
DEF KEY( 1)="RUN␣␣␣"+CHR$(13)
DEF KEY( 2)="LIST␣"
DEF KEY( 3)="AUTO␣"
DEF KEY( 4)="RENUM␣"
DEF KEY( 5)="COLOR␣"
DEF KEY( 6)="CHR$("
DEF KEY( 7)="DEF␣KEY("
DEF KEY( 8)="CONT"
DEF KEY( 9)="SAVE␣␣"
DEF KEY(10)="LOAD␣␣"
Ready
```

Die obige Liste enthält die ursprünglichen Einstellungen für die definierbaren Funktionstasten.

Hinweis:
"␣" bedeutet eine Leerstelle.

INIT

Format

(1) **INIT** "RAM:[<Anzahl der Bytes>]"

(2) **INIT** "LPT:[M {0, 1, 2}] [, Sn] [, CR-Code]"

(3) **INIT** "RS {1, 2}: [<Überwachungscode>, <Initialisiercode>[, <Endcode>]"

(4) **INIT** "CRT: [M<Betriebsart>][,B<Blockcode>]"

Erklärung

Der INIT-Befehl dient zur Festlegung der Anfangseinstellungen und Betriebsarten für externe Geräte.

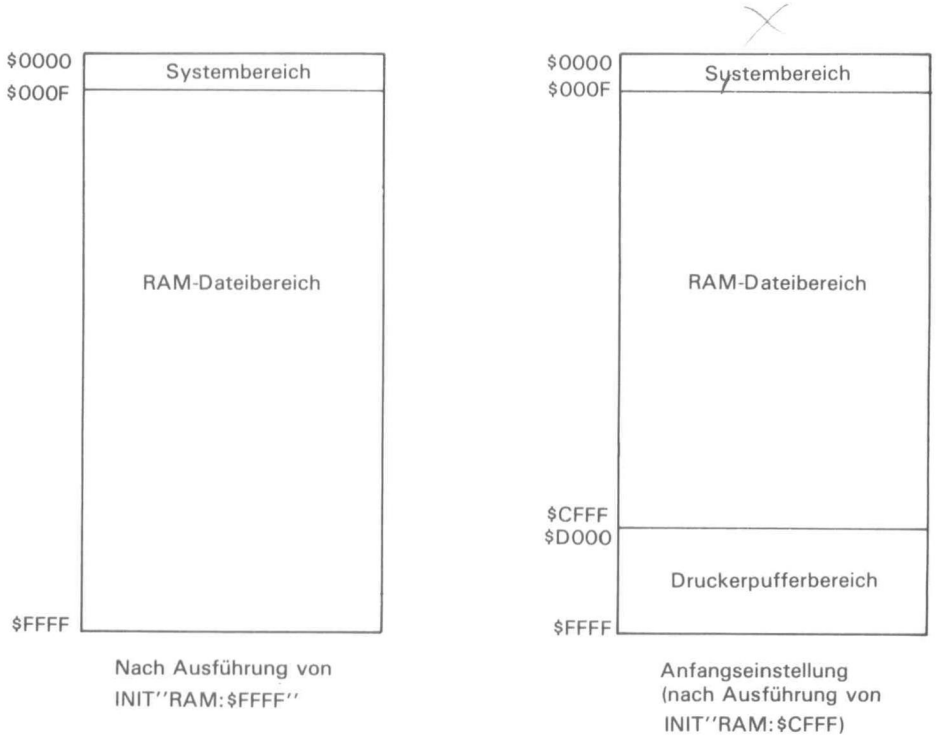
Format 1

In diesem Format initialisiert der INIT-Befehl die optionale RAM-Datei-Steckkarte (MZ-1R18), weist die in <Anzahl der Bytes> spezifizierte Speicherplatz-Größe dieser Datei und den verbleibenden Speicherplatz dem Druckerpuffer zu.
Die <Anzahl der Bytes> muß sich innerhalb des Bereiches von $0010 bis $FFFF befinden. Wenn die Angabe <Anzahl der Bytes> ausgelassen wird, so wird die gegenwärtige Einstellung für den RAM-Dateibereich vorausgesetzt. Bei Ausführung dieses Befehls erscheint die Meldung "OK? [Y/N]". Gibt man nun Y ein, wird der RAM-Dateibereich wie nachstehend gezeigt festgelegt. Bei Eingabe von N erfolgt Anzeige der "Break" Meldung und Rückkehr in die Direktbefehls-Betriebsart.
Wenn man der RAM-Datei-Steckkarte oder dem Druckerpuffer zu wenig Speicherplatz zuweist, steht die Komponente möglicherweise nicht zur Verfügung.

Beispiel

```
INIT "RAM:$FFFF"
```

Dieser Befehl intialisiert die RAM-Datei-Steckkarte und weist der RAM-Datei die maximale Speicherplatz-Größe zu.



Format 2

In diesem Format wird mit dem INIT-Befehl der Drucker und die Betriebsart, in der der Druckerpuffer verwendet werden soll, spezifiziert.

[M] zeigt die Betriebsart an, in der der Druckerpuffer verwendet wird.
M0: Direkt-Betriebsart (Der Puffer wird initialisiert)
M1: Spool-Betriebsart (Der Puffer wird initialisiert)
M2: Direkt-Betriebsart (Wenn zur Zeit die Spool-Betriebsart aktiviert ist, wird nach Ausdrucken des Pufferinhalts diese Betriebsart aktiviert.)

Die Optionen M1 und M2 sind nur gültig, wenn eine RAM-Datei-Steckkarte eingesetzt ist.
Wenn in der Spool-Betriebsart der Bilddruckcode 0BH + 0BH* an den Drucker MZ-80P5(K) übertragen wird, tritt ein Fehler auf. Zur Aufhebung eines derartigen Fehlerzustands gibt man den gewünschten Befehl nach Ausführung von INIT"LPT:M2" nochmals ein. In der Spool-Betriebsart läßt sich der Druckvorgang stoppen, indem man die Tasten [CTRL] und [N] gleichzeitig drückt.

* H zeigt an, daß die vorangestellte Zahl eine Hexadezimalzahl ist.

[S] spezifiziert den Drucker.
S0: MZ-IP16
S1: MZ-80P5(K)
S2: Drucker, der Druckdaten in ASCII Code umwandelt.
S3: Durchgehender Code

Die folgenden Codes werden während der Ausführung des PRINT/P Befehls wie nachstehend gezeigt umgewandelt, wenn im INIT Befehl S0 oder S1 spezifiziert ist.

CHR$ ($11) oder [↓] wird in $09 umgewandelt.
CHR$ ($12) oder [↑] wird in $0B umgewandelt.
CHR$ ($15) oder [H] wird in $0F umgewandelt.
CHR$ ($16) oder [C] wird in $0C und $0A umgewandelt.

Wenn ein anderer Code als 0DH als der CR-Code verwendet werden soll, muß der <CR Code> spezifiziert werden.

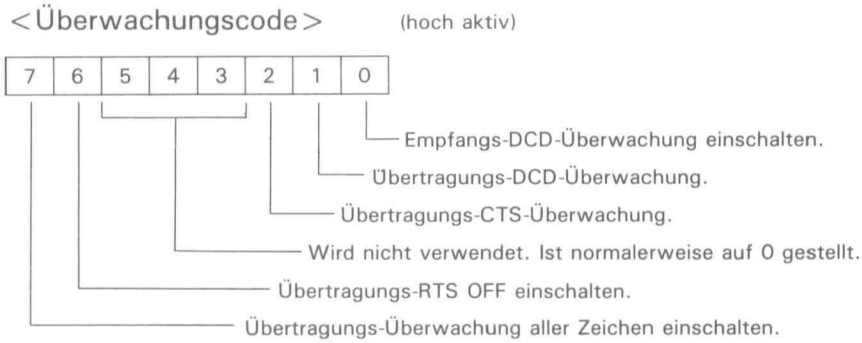
Beispiel

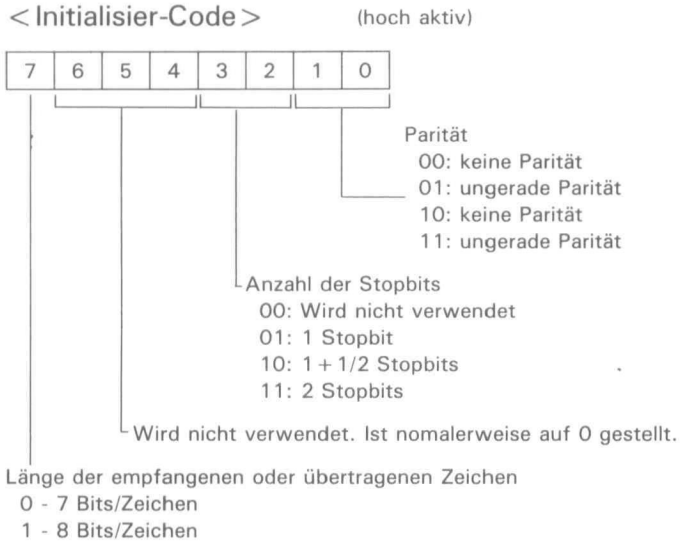
```
INIT "LPT: M1, S1"
```

Im obigen Beispiel wird spezifiziert, daß ein Teil des RAM-Bereichs als Druckerpuffer verwendet werden soll und als Druckerpuffer-Betriebsart wird Spool vorgegeben. Der zu verwendende Drucker ist der MZ-80P5(K).

Format 3

In diesem Format dient der INIT Befehl zur Vorgabe der Betriebsart, in der die Schnittstelle RS-232C zu betreiben ist.





<Endcode>

Eine Zahl von 0 bis 255 ($00 — $FF)

• Der Druckerpuffer

Bei Ausgabe der Daten an den Drucker wartet der Computer, bis alle Daten gedruckt sind, bevor er dann die Verarbeitung vornimmt. Da die Druckgeschwindigkeit jedoch wesentlich langsamer ist als die Verarbeitungsgeschwindigkeit des Computers, verbringt der Computer den größten Teil der Zeit einfach damit, darauf zu warten, daß der Drucker fertig wird. Wenn man die Daten jedoch in einen speziellen Zwischenspeicher eingibt und dieser die Daten an den Drucker überträgt, sobald der Drucker fertig (wieder einsatzbereit) ist, braucht der Computer nicht auf den Drucker zu warten, stattdessen kann man ihn für andere Zwecke verwenden, während der Drucker ausdruckt. Einen solchen speziellen Zwischenspeicher nennen wir Druckerpuffer.

Mit dem MZ-800 kann ein Teil des Speichers der als Option erhältlichen RAM-Datei [MZ-1R18] als Druckerpuffer verwendet werden. Der Druckerpuffer-Betrieb wird aktiviert, wenn M1 spezifiziert wird (die Spool-Betriebsart), wird mit dem Format 2 des INIT-Befehls spezifiziert und aufgehoben, wenn M0 oder M2 (die Direkt-Betriebsart) spezifiziert wird.

Beispiel

Für die Datenübertragung zwischen zwei MZ-800 ist, wie nachstehend gezeigt, ein Anschlußkabel für die RS-232C Buchsen vorzubereiten.

Signalname	Stiftnummer		Stiftnummer	Signalname
TXD	2		2	TXD
RXD	3		3	RXD
RTS	4		4	RTS
CTS	5		5	CTS
DTR	6		6	DTR
DCD	7		7	DCD
Erde	1, 8		1, 8	Erde

Beide Geräte in der Terminal-Betriebsart verbinden. (Hinweise zur Schnittstelle RS-232C siehe Handbuch)
Die folgenden Programme dienen zur Übertragung/zum Empfang des Inhalts von A$ an den/vom angeschlossenen MZ-800.

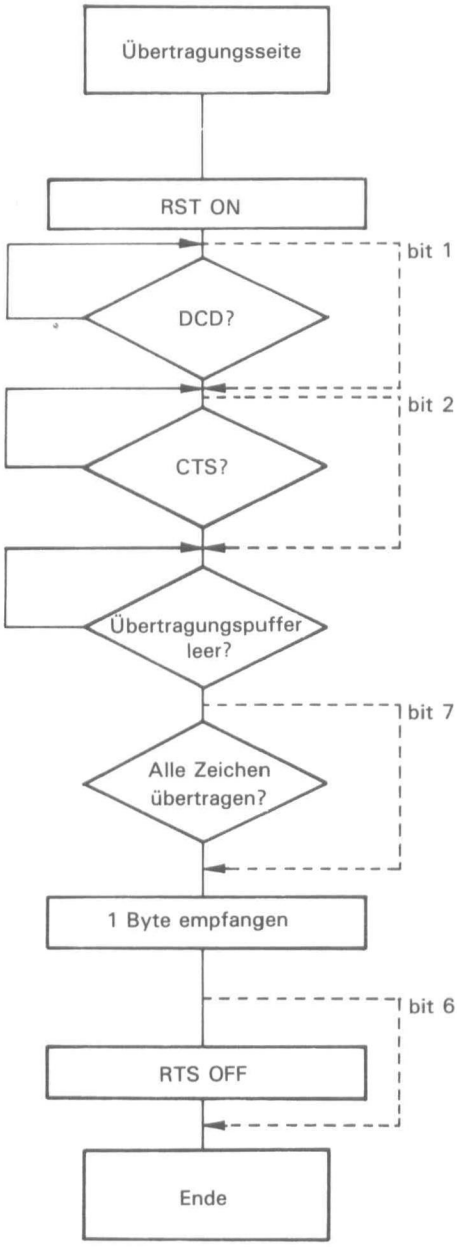
[Programm für Übertragungsseite]

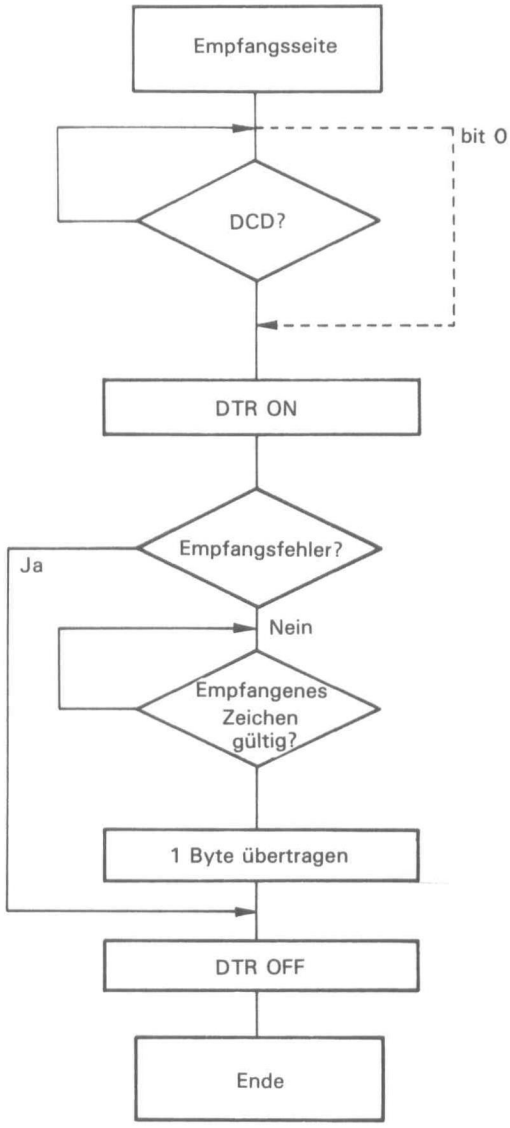
```
10 INIT''RS1:$00,$8C''
20 A$=''0123456789''
30 WOPEN#1,''RS1:''
40 PRINT#1,A$
50 CLOSE#1
60 END
```

[Programm für Empfangsseite]

```
10 INIT''RS1:$00,$8C''
20 ROPEN#2,''RS1:''
30 INPUT#2,A$
40 PRINT A$
50 CLOSE#2
60 END
```

<Programmablaufplan>





Format 4

In diesem Format dient der INIT-Befehl zur Vorgabe der Anzeige-Einstellungen. Die <Betriebsart> spezifiziert die Auflösung des Bildschirms und die Anzahl der Farben wie folgt.

Betriebs-art	Auflösung	Farben	Zeichen/Zeile
1	320 × 200	4	40
2	320 × 200	16	40
3	640 × 200	einfarbig	80
4	640 × 200	4	80

Bei Verwendung eines Fernsehgerätes wird in den Betriebsarten 3 und 4 keine ausreichende Auflösung erzielt.

<Blockcode> spezifiziert die Farbpaletten-Blocknummer. ($0 \leqq$ <Blockcode> $\leqq 3$).

BYE

Format **BYE**

Abgekürztes Format

B.

Erklärung Mit dem BYE -Kommando erfolgt die Rückkehr vom BASIC Interpreter zu dem im RAM stehenden Monitorprogramm.
Für weitere Einzelheiten zum Monitorprogramm siehe Kapital 8.

BOOT

Format **BOOT**

Erklärung Das BOOT-Kommando startet den Urlader (IPL). Der Computer wird dadurch in den gleichen Zustand versetzt wie beim Einschalten des Gerätes.

Beispiel BOOT ..... Lädt das Systemprogramm nochmals in den Speicher.

WAIT

Format **WAIT** <numerische Daten>

Abgekürztes Format

W.

Erklärung Die WAIT-Anweisung unterbricht die Programmausführung für die durch <numerische Daten> spezifizierte Dauer. Die Zeit ist in Millisekunden (1/1000 sec.) zu spezifizieren.

Beispiel WAIT 100 ..... unterbricht die Programmausführung 0.1 (100/1000) sec lang.

6.3 Dateisteuerbefehle

DIR

Format

DIR[/P][RAM]

Erklärung

Das DIR-Kommando zeigt die Namen der mit der RAM-Datei-Steckkarte gespeicherten Dateien an.
Durch Angabe von DIR/P wird der Inhalt vom Dateienverzeichnis an den Drucker ausgegeben.
Die Angabe RAM kann auch ausgelassen werden, wenn die RAM-Datei-Steckkarte im DEFAULT-Kommando spezifiziert ist oder die Standardvorgabe sie schon als das angemeldete Gerät vorsieht. Dieses Kommando ist nur gültig, wenn die optionale RAM-Datei-Steckkarte MZ-IR18 installiert ist.
Jedem Datei-Namen ist einer der folgenden 3 Datei-Typen nachgestellt.

BTX: BASIC-Programmdateien
BSD: BASIC-Datendateien (Serielle Dateien)
OBJ: Maschinensprache-Dateien

Beispiel

DIR RAM..... das Dateienverzeichnis der RAM-Datei-Steckkarte wird angezeigt.

Siehe hierzu auch

DEFAULT

RUN

Format **RUN** ["[<Gerätename>:]<Dateiname>"[,{A|R}]]

Abgekürztes Format

R.

Erklärung Löscht die vorhandenen Programme im BASIC-Programmbereich und löscht den Programmarbeitsbereich, lädt daraufhin das durch <Dateiname> spezifizierte Programm von dem durch <Gerätename> ausgewiesenen Gerät in den BASIC-Programmbereich. Daraufhin bewirkt dieser Befehl die Ausführung des Programms vom Anfang an. Wenn das Standardvorgabe-Gerät oder das im DEFAULT-Befehl spezifizierte Gerät verwendet werden soll, kann die Angabe <Gerätename> auch ausgelassen werden. Wenn man diesen Befehl ganz ohne Parameter eingibt, wird das im BASIC-Programmbereich stehende Programm nicht gelöscht.
Zum Laden und zur Ausführung eines Programms, das in Form einer BSD-Datei (im ASCII-Code aufgezeichnet) ist, spezifiziert man die A-Option.
Durch Vorgabe der R-Option wird das Laden eines OBJ-Programms in der gleichen Weise wie das Laden des IPL ermöglicht.
Die ladbaren Dateitypen sind BTX, BSD und OBJ.

Beispiel

RUN "CMT:PROG" ..... Die BTX-Datei "PROG" wird von der Cassette geladen und ausgeführt.

RUN "CMT:DATEN",A ..... Die BSD-Datei "DATEN" wird von der Cassette geladen und ausgeführt.

LOAD

Format

LOAD "[<Gerätename>:]<Dateiname>" [,A]

Abgekürztes Format

LO.

Erklärung

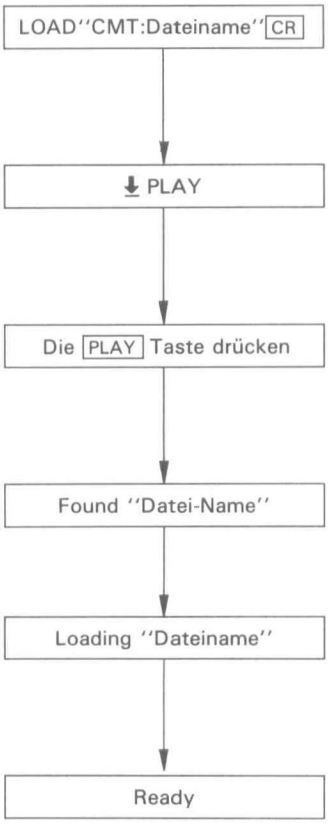
Der LOAD-Befehl dient zum Laden des angegebenen Programms von einem externen Speichergerät in den Speicher.
Beim Laden wird als <Dateiname> der gleiche Name verwendet, unter dem die Datei zuvor mit dem externen Speichergerät gesichert (gespeichert) wurde. Die Angabe des folgenden Parameters ist nicht unbedingt erforderlich. (Gerätename) ist entweder CMT oder RAM. Dieser Parameter kann auch ausgelassen werden, wenn das Standardvorgabe-Gerät oder das im DEFAULT-Kommando spezifizierte Gerät zu verwenden ist.
Zum Laden einer im ASCII-Format gespeicherten BSD-Datei ist außerdem die Eingabe der A-Option erforderlich. Bitte beachten, daß das Einlesen von Dateien im ASCII-Format länger dauert als das Einlesen von Dateien im Binärformat.
Mit diesem Befehl können nur BASIC-Programmdateien und Maschinensprache-Programme geladen werden. Wenn die zu ladende Datei eine BASIC-Programmdatei ist, wird durch Laden des neuen Programms das zur Zeit im BASIC-Textbereich stehende Programm gelöscht.

Hinweis:
Für das Laden eines Maschinensprache-Unterprogramms, das mit dem BASIC-Programm verknüpft ("gelinkt") werden soll, ist der LIMIT-Befehl auszuführen, um im Speicher einen Maschinenspracheprogramm-Bereich zu reservieren. Ferner gilt, daß das betreffende Maschinenspracheprogramm sofort nach dem Laden ausgeführt wird, wenn die Ladeadresse sich innerhalb dieses Bereichs befindet. (Der BASIC-Text wird in diesem Fall nicht gelöscht) Der LOAD-Befehl kann auch innerhalb eines Programms zum Laden eines Maschinenspracheprogramms verwendet werden.

Beispiel

LOAD "CMT:HELLO" ..... lädt eine Datei mit Namen "HELLO" von der Cassette.

Verfahrensweise zum Laden eines Programms



Eingeben

Bildschirmanzeige

Datenrecorder-Bedienung

Bildschirmanzeige (Der Name des gefundenen Programms wird angezeigt)

Bildschirmanzeige

Bildschirmanzeige

SAVE

Format

SAVE "[<Gerätename>:]<Dateiname>"[,A]
<Gerätename> muß CMT oder RAM sein.

Abgekürztes Format

SA.

Erklärung

Mit dem SAVE-Befehl wird das im Arbeitsspeicher stehende BASIC-Programm unter dem spezifizierten Dateinamen auf dem externen Datenträger gespeichert.
Wenn es sich um das Gerät handelt, das mit der Standardvorgabe oder einem DEFAULT-Befehl bereits spezifiziert ist, kann die Angabe des Parameters <Gerätename:> auch ausgelassen werden.
Der Parameter <Dateiname> ist erforderlich und muß spezifiziert werden. Durch zusätzliche Angabe der A-Option speichert der SAVE-Befehl das BASIC-Programm im ASCII-Format. In diesem Falle wird die Datei vom BASIC-Interpreter mit dem Zusatz BSD versehen. Mit dem SAVE-Befehl können Dateien vom Typ BTX und BSD gespeichert werden.

Beispiel

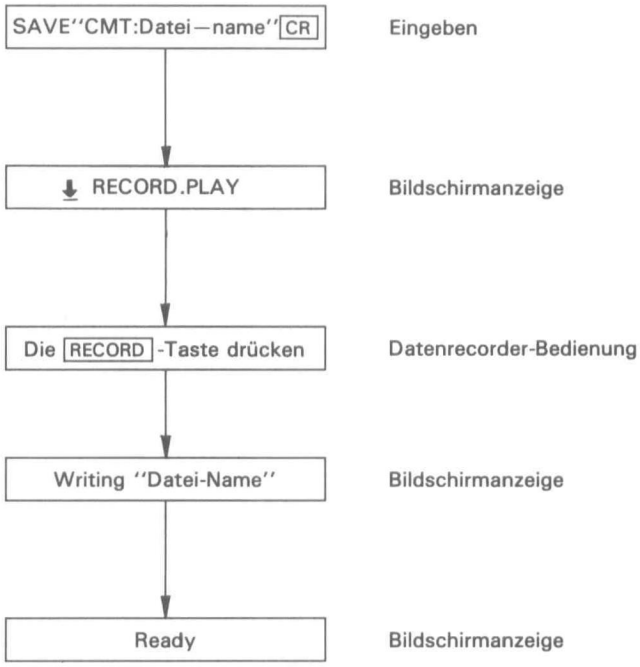
SAVE"CMT:PROG"..... sichert das im Speicher stehende Programm unter dem Dateinamen "PROG" auf Cassette. Als Dateityp des gespeicherten Programms wird BTX vorausgesetzt.

SAVE"CMT:DATEN",A..... sichert das im Speicher stehende Programm im ASCII-Format unter dem Dateinamen "DATEN" auf Cassette. Als Dateityp des gespeicherten Programms wird BSD vorausgesetzt.

Hinweis:

Der SAVE-Befehl kann nur zum Speichern von BASIC-Programmen (d.h., von Programmen, die durch Ausführung des LIST-Befehls angezeigt werden können) verwendet werden; er ist nicht zum Speichern von Maschinensprache-Programmen im Maschinensprach-Bereich geeignet.
Bei Verwendung von SAVE empfiehlt es sich, den Stand des Bandzählwerks für spätere Bezugnahme zu notieren.

Verfahrensablauf zum Speichern eines BASIC-Programms



VERIFY

Format **VERIFY**"[CMT]: <Dateiname>"

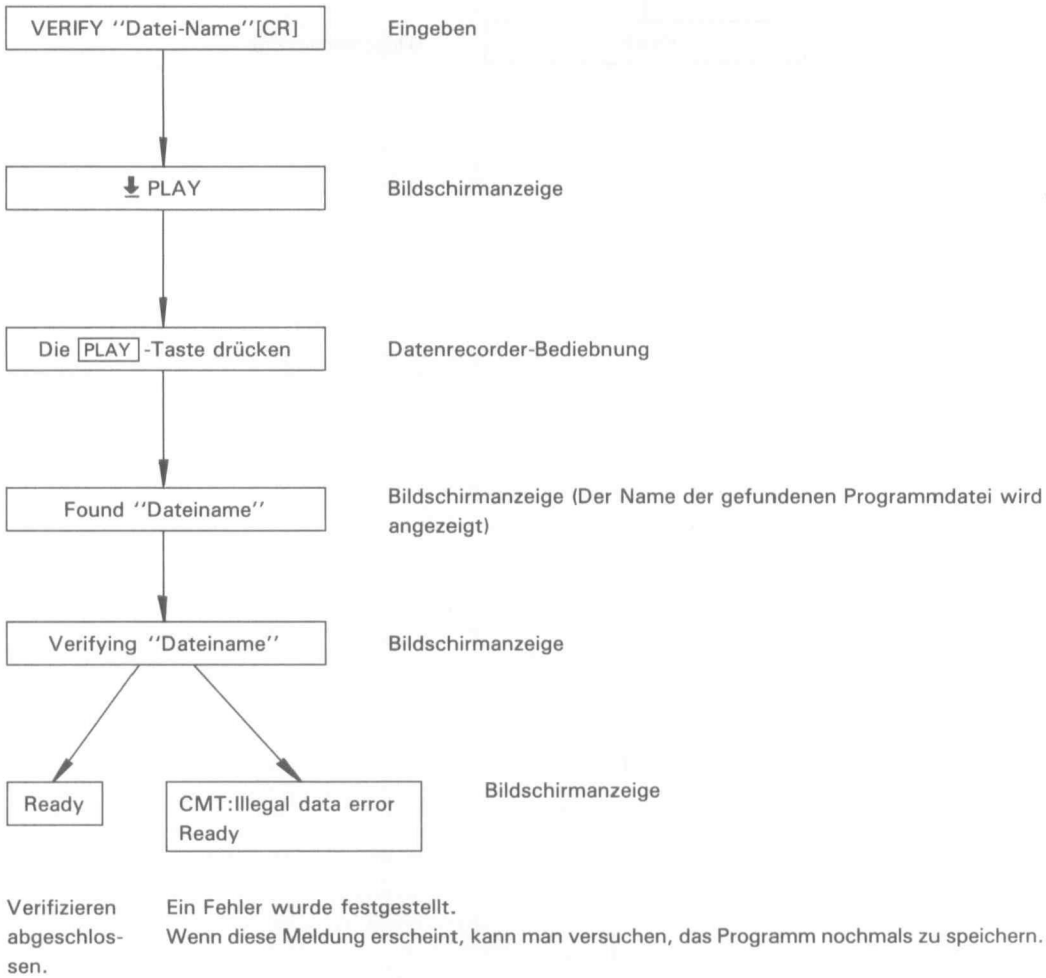
Abgekürztes Format

V.

Erklärung Der VERIFY-Befehl vergleicht das im Speicher stehende Programm mit dem auf Cassette geschriebenen Programm; er dient zur Bestätigung, daß das Programm ordnungsgemäß gespeichert wurde. Wenn beide Programme gleich sind, wird "Ready" angezeigt, falls Unterschiede festgestellt werden, erscheint die Anzeige "CMT:Illegal data error". Im letzteren Falle nimmt man das Speichern des Programm nochmals vor. Dieser Befehl ist nur für Dateien auf Cassette gültig.

Beispiel VERIFY "CMT:NAME"..... vergleicht die Datei "NAME" auf Cassette mit dem Programm im Speicher.

Verfahrensschritte zum Verifizieren einer Programmdatei.



DELETE

Format **DELETE**"[RAM:] <Dateiname>"

Abgekürztes Format

D.

Erklärung Der DELETE-Befehl löscht die durch <Dateiname> spezifizierte Datei. Dieser Befehl kann nur ausgeführt werden, wenn die optionale RAM-Datei-Steckkarte MZ-1R18 eingesetzt ist.

Beispiel DELETE "RAM:BEISPIEL".... löscht die auf der RAM-Datei-Steckkarte befindliche Datei namens "BEISPIEL".

RENAME

Format **RENAME**"[RAM:]<alter Dateiname>","<neuer Dateiname>"

Abgekürztes Format

RENA.

Erklärung Der RENAME-Befehl dient zur Umbenennung einer bestehenden Datei. Zur Umbenennung einer bestehenden Datei spzifiziert man den alten und den neuen Dateinamen in dieser Reihenfolge. Wenn der angegebene neue Dateiname bereits im Dateienverzeichnis auf der RAM-Datei-Steckkarte enthalten ist, tritt ein Fehler auf. Die Angabe RAM kann auch ausgelassen werden, wenn die RAM-Datei-Steckkarte als Standardvorgabe bestimmt oder mit dem DEFAULT-Befehl als Standardvorgabe-Gerät spezifiziert ist.
Zur Ausführung des RENAME-Befehls ist die optionale RAM-Datei-Steckkarte MZ-1R18 erforderlich.

Beispiel RENAME"RAM:ALTPROG","NEUPROG"

In diesem Beispiel wird der Name einer auf der RAM-Datei-Steckkarte enthaltenen Datei von "ALTPROG" nach "NEUPROG" umbenannt.

CHAIN

Format

CHAIN"[<Gerätename>:]<Dateiname>"
<Gerätename> muß CMT oder RAM sein.

Abgekürztes Format

CH.

Erklärung

Der CHAIN-Befehl überträgt die Ablaufsteuerung vom gegenwärtigen Programm zu einem anderen, in einer Datei befindlichen Programm. Ferner hat der CHAIN-Befehl die Funktion, eine Datei zu öffnen. Die Ausführung eines CHAIN-Befehls hat die gleiche Wirkung wie die Ausführung eines RAM-Befehls in einem Programm, mit dem Unterschied das CHAIN die Werte von Variablen und vom gegenwärtigen Programm an das aufgerufene Programm weitergibt.
Wenn das als Standardvorgabe eingestellte oder das mit dem DEFAULT-Befehl spezifizierte Gerät verwendet werden soll, kann die Angabe <Gerätename> auch ausgelassen werden.

Beispiel

```
10 A=1
20 B=2
30 CHAIN "CMT:PROG"
40 END
```

In diesem Beispiel-Programm wird die Ablaufsteuerung in Zeile 30 an das unter dem Dateinamen"PROG" auf Cassette gespeicherte Programm übertragen. Die Werte der Variablen A und B, 1 und 2 werden an das aufgerufene Programm übergeben.

MERGE

Format

MERGE["[<Gerätename>:]<Dateiname>"][,A]
<Gerätename> muß CMT oder RAM sein.

Abgekürztes Format

M.

Erklärung

Der MERGE-Befehl vereint das durch <Dateiname> spezifizierte Programm mit dem gegenwärtig im Speicher stehenden.
Bei Verwendung des als Standardvorgabe eingestellten oder des mit dem DEFAULT-Befehl spezifizierten Gerätes kann die Angabe <Gerätename> auch ausgelassen werden.
Wenn Zeilennummern des zugemischten Programms mit Zeilennummern des im Speicher stehenden Programms übereinstimmen, werden die entsprechenden Zeilen des im Speicher stehenden Programms durch die Zeilen des neu eingelesenen Programms überschrieben.
Um den MERGE-Befehl für ein im ASCII-Format gespeichertes Programm (BSD-Dateityp) zu verwenden, ist es erforderlich, am Ende des Befehls die A-Option einzugeben.

Beispiel

(Programm im Speicher)	(Programm auf Cassette) "PROG"
10 B=2	10 A=1
30 PRINT B	20 PRINT A
50 END	40 END

Wenn diese beiden Programme unter Verwendung des MERGE "CMT:PROG"-Befehls gemischt werden, sieht das gemischte Programm wie folgt aus:

```
10 A=1
20 PRINT A
30 PRINT B
40 END
50 END
```

Verwenden Sie den LIST-Befehl zur Bestätigung.

WOPEN

Format

WOPEN # <Dateinummer>,"[<Gerätename>:] <Dateiname>"
<Dateinummer> muß eine Ganzzahl von 1 bis 127 sein.
<Gerätename> muß CMT, RAM oder RSn sein.

Abgekürztes Format

WO. #

Erklärung

Der WOPEN #-Befehl öffnet eine BSD-Datei für die Ausgabe. Desgleichen wird der Datei eine Dateinummer und ein Dateiname zugewiesen.
Bei Verwendung des Standardvorgabe-Gerätes oder des mit dem DEFAULT-Befehl spezifizierten Gerätes kann die Angabe <Gerätename> auch ausgelassen werden.
Wenn man RSn als <Gerätename> spezifiziert, erfolgt die Ausgabe an das über die Schnittstelle RS-232C angeschlossene Gerät.

Beispiel

```
10 WOPEN #1,"CMT:DATEN" ...
```
Eröffnet eine Datei unter dem Namen "DATEN" zur Ausgabe und weist ihr die Dateinummer 1 zu.

```
10 WOPEN #1, "RS1:"..........
```
Nach Ausführung dieses Befehls werden alle Ausgaben der PRINT #1 Anweisung an die Schnittstelle RS-232C übertragen.

```
10 WOPEN #2, "DATEN"
20 FOR Z=1 TO 99
30 PRINT #2,Z
40 NEXT Z
50 CLOSE #2
60 END
```

Das obige Beispielprogramm schreibt 1 bis 99 in die spezifizierte Datei.

Siehe hierzu auch

PRINT #, ROPEN #, CLOSE #

PRINT

Format

PRINT # <Dateinummer>,<Daten>[,<Daten>]...

Abgekürztes Format

? #

Erklärung

Die PRINT # Anweisung schreibt Daten sequentiell in die Datei, die mit der WOPEN # Anweisung eröffnet wurde.
Als <Dateinummer> ist die gleiche Nummer zu verwenden wie bei der Eröffnung der Datei zur Ausgabe mit der WOPEN #-Anweisung. Die <Daten> können numerisch oder alphanumerisch sein.

Beispiel

```
10 WOPEN #1, "CMT: DATEN2"
20 PRINT #1, 1, 2, 3
30 CLOSE #
40 END
```

Dieses Beispielprogramm schreibt die numerischen Daten 1,2 und 3 unter der Dateinummer 1 in die Datei "DATEN 2" die für Ausgabe eröffnet wurde.

Siehe hierzu auch

WOPEN #, CLOSE #

ROPEN

Format

ROPEN # <Dateinummer>, "[<Gerätename>:]<Dateiname>"
<Dateinummer> muß eine Ganzzahl von 1 bis 127 sein.
<Gerätename> muß CMT, RAM oder RSn sein.

Abgekürztes Format

RO. #

Erklärung

Die ROPEN #-Anweisung öffnet eine Datei für die Eingabe. Mit der ROPEN #-Anwelsung wlrd der durch <Gerätename> und <Dateiname> bestimmten Datei eine <Dateinummer> zugewiesen. Die <Dateinummer> muß eine Ganzzahl von 1 bis 127 sein.
Bei Verwendung des Standardvorgabe-Gerätes oder des mit dem DEFAULT-Befehl spezifizierten Gerätes kann die Angabe <Gerätename> auch ausgelassen werden. Wenn als <Gerätename> "RSn:" spezifiziert ist, wird die Schnittstelle RS-232C als das Eingabegerät bestimmt, von dem die Daten zu lesen sind.

Beispiele

10 ROPEN #1, "CMT: DATEN".... Öffnet eine BSD-Datei namens "DATEN" auf Cassette.

10 ROPEN #1, "RS1:" ..... Bestimmt das an die Schnittstelle RS-232C angeschlossene Gerät als das Gerät, von dem alle in INPUT #1 Anweisungen spezifizierten Daten zu lesen sind.

```
10 ROPEN #2, "DATEN"
20 FOR Z=1 TO 99
30 INPUT #2,A
40 PRINT A
50 NEXT Z
60 CLOSE #
70 END
```

Das obige Programm liest den Inhalt der Datei, die durch das für die WOPEN # Anweisung gegebene Beispielprogramm geschaffen wurde und zeigt diesen auf dem Bildschirm an.

Siehe hierzu auch

INPUT #, WOPEN #, CLOSE #

INPUT

Format

INPUT # <Dateinummer>,<Variable>[,<Variable>]...

Abgekürztes Format

I. #

Erklärung

Die INPUT # Anweisung dient dazu, Daten aus der mit der ROPEN #-Anweisung geöffneten Datein sequentiell zu lesen und sie den Programmvariablen zuzuweisen. <Variable> kann auch ein Feldvariablenelement sein. Die <Dateinummer> ist die gleiche Nummer, die zum Öffnen der Datei für die Eingabe mit der ROPEN #-Anweisung verwendet wurde.
Genau wie beim READ ~ DATA-Anweisungspaar kann ein Fehler entstehen, wenn Daten- und Variablentyp nicht übereinstimmen. Wenn die spezifizierte Datei sich auf der RAM-Datei-Steckkarte befindet, kann man die EOF #-Funktion verwenden, um zu überprüfen, ob das Dateiende erreicht ist.

Beispiel

```
10 ROPEN #2, "DATEN2"
20 INPUT #2, A, B, C
30 PRINT A, B, C,
40 CLOSE #2
50 END
```

Dieses Beispielprogramm liest die numerischen Daten aus der unter der Dateinummer 2 für Eingabe geöffneten Datei und weist die Daten den numerischen Variablen A, B und C zu.

Siehe hierzu auch

ROPEN #, CLOSE #, EOF #

EOF(#)

Format

EOF(# Dateinummer >)

Abgekürztes Format

EO. #

Erklärung

Die EOF # -Funktion dient zum Auffinden des Endes einer Datei. Wenn alle in der Datei enthaltenen Daten gelesen sind, signalisiert diese Funktion den Dateiende-Zustand. Nachdem das Dateiende festgestellt wurde, wird der angegebene Wert – 1 (wahr). Beim Lesen der Daten vom Cassettenrecorder ist EOF (#) ungültig.
Die EOF (#) Funktion wird im allgemeinen zusammen mit der IF-Anweisung verwendet; man plaziert sie möglichst hinter einer INPUT # -Anweisung.

Beispiel

```
10 ROPEN #3, "DATEN"
20 INPUT #3, A
30 IF EOF( #3) THEN END
40 PRINT A
50 GOTO 20
```

Das obige Programm liest Daten sequentiell von einer Datei namens "DATEN" und zeigt sie auf dem Bildschirm an, bis das Dateiende festgestellt wird.

Siehe hierzu auch

INPUT #

CLOSE

Format **CLOSE**[# <Dateinummer>]

Abgekürztes Format

CLO. #

Erklärung: Mit der CLOSE-Anweisung wird die unter der spezifizierten Dateinummer geöffnete Datei geschlossen. Nach Ausführung einer CLOSE-Anweisung wird die bisher der Datei zugewiesene Dateinummer freigegeben.
Eine CLOSE-Anweisung bezogen auf eine für Ausgabe eröffnete Datei bewirkt das Räumen des Ausgabepuffers. Wenn der CLOSE-Anweisung keine Dateinummer zugewiesen ist, werden alle offenen Dateien geschlossen und alle Dateinummern freigegeben.

Beispiel

10 CLOSE # 1 ..... Schließt die Datei, der die Dateinummer 1 zugewiesen ist.

10 CLOSE ..... Schließt alle offenen Dateien.

Siehe hierzu auch

WOPEN #, ROPEN #

KILL

Format **KILL**[# <Dateinummer>]

Abgekürztes Format

KI. #

Erklärung Der KILL # -Befehl bricht das Schreiben von Daten in die unter der spezifizierten Dateinummer eröffnete Datei ab. Ein KILL # -Befehl ohne Angabe der Dateinummer bewirkt den Abbruch des z.Z. stattfindenden Schreibbetriebs, schließt alle offenen Dateien und gibt die Dateinummern frei.

Beispiel

KILL # 3 ..... Bewirkt Abbruch des Schreibens in die unter der Dateinummer 3 geöffnete Datei und gibt die der Datei zugewiesene Dateinummer frei.

Siehe hierzu auch

WOPEN #, PRINT #

DEFAULT

Format

DEFAULT "<Gerätename>:"

Abgekürztes Format

DEF.

Erklärung

Der DEFAULT-Befehl bestimmt den Gerätenamen, der vorauszusetzen ist, wenn die Angabe des Parameters <Gerätename> in Eingabe/Ausgabe-Anweisungen ausgelassen wird.
Die Gerätenamen werden wie folgt spezifiziert:
CMT..... Datenrecorder (Standardvorgabe)
RAM..... RAM-Datei-Steckkarte
LPT..... Drucker
RS$\left\{\begin{matrix}1\\2\end{matrix}\right\}$..... Schnittstelle RS-232C

Beispiel

```
DEFAULT "CMT:"
```

Nach Ausführung dieser Anweisung wird stets der Cassettenrecorder als Gerät vorausgesetzt, wenn der Parameter <Gerätename> in Eingabe/Ausgabe-Anweisungen für externe Geräte ausgelassen wird.

6.4 Grafik-Steuerbefehle

COLOR

Format

COLOR [<Palettencode>[, <Betriebsart>]

Abgekürztes Format

COL.

Erklärung

Die COLOR-Anweisung spezifiziert den <Palettencode> und die optionale <Betriebsart>, die durch PRINT, PRINT USING und die Grafik-Anweisungen SET, RESET, LINE, BLINE, BOX, CIRCLE, PAINT, PATTERN und SYMBOL verwendet werden.
Die <Betriebsart> spezifiziert den Typ der logischen Summe von Farben. Wenn als <Betriebsart> 0 spezifiziert ist, werden die alten Farben in den überlagerten Abschnitten durch die neuen Farben ersetzt. Wenn dieser Parameter als 1 spezifiziert ist, kommt ein logisches ODER der alten und der neuen Farben zur Anwendung. Dies gilt jedoch nicht für RESET und BLINE. Siehe Anhang A.

Beispiel

```
10 INIT "CRT:M1"
20 COLOR 3,0
30 FOR J=0 TO 10 STEP
40 SET 100,J
50 NEXT J
60 END
```

Dieses Programm plottet Punkte an den Stellen (100, 0) und (100, 2) bis (100, 10) in den Farben gemäß Palettencode 3 und in der Überlagerungs-Betriebsart 0.

Siehe hierzu auch

Anhang A.

PAL

Format	**PAL** <Palettencode>,<Farbcode>
Erklärung	Mit der PAL-Anweisung wird der Zusammenhang zwischen einem Palettencode und Farbcodes hergestellt. Sowohl Palettencode-als auch Farbcode - Parameter können Werte von 0 bis 15 annehmen. In allen anderen Farben-Betriebsarten als der 16-Farben-Betriebsart kann der Benutzer zwei oder vier Palettencodes und 16 Farben gleichzeitig wählen. In der 16-Farben-Betriebsart kann der Benutzer mit dem INIT-Befehl einen Palettenblock einrichten und für diesen Block vier Paletten wählen, so daß insgesamt 16 Farben gewählt werden können. Nachstehend sind die Standardvorgaben (Anfangswerte) der Paletten- und Farbcodes aufgeführt.

(1) 2-Farben-Betriebsart

Palettencode	Farbcode
0	0 Schwarz
1	15 Hellweiß

(2) 4-Farben-Betriebsart

Die nachstehende Tabelle zeigt das Verhältnis zwischen Paletten- und Farbcode, das beim Starten von BASIC etabliert wird.

Palettencode	Farbcode
0	0 Schwarz
1	1 Blau
2	2 Rot
3	15 Hellweiß

Man kann aus 16 Farbcodes vier Farbcodes auswählen.

(3) 16-Farben-Betriebsart

Die Palettencode-Standardvorgabewerte sind mit denen in der Farb-Betriebsart identisch. In dieser Betriebsart sind die Farben für jeden einzelnen Palettenblock festgelegt (siehe "INIT-Befehl" für Palettenblöcke).
n: Palettenblocknummer

n	Farbcode	Farbe
0	0	Schwarz
	1	Blau
	2	Rot
	3	Magenta
1	4	Grün
	5	Cyan
	6	Gelb
	7	Weiss

n	Farbcode	Farbe
2	8	Grau
	9	Hellblau
	10	Hellrot
	11	Hellmagenta
3	12	Hellgrün
	13	Hellcyan
	14	Hellgelb
	15	Hellweiß

Hinweis:
Wenn ein Palettenblock mit dem INIT-Befehl in der 16-Farben-Betriebsart verändert wird, bleibt die vor dem Wechsel eingerichtete Palette für den nächsten Block gültig. Andere Farben als die zum Palettenblock gehörenden werden in die grundlegenden abgewandelt. Für weitere Einzelheiten zu Farbcodes und Palettencodes siehe Anhang A.

SET

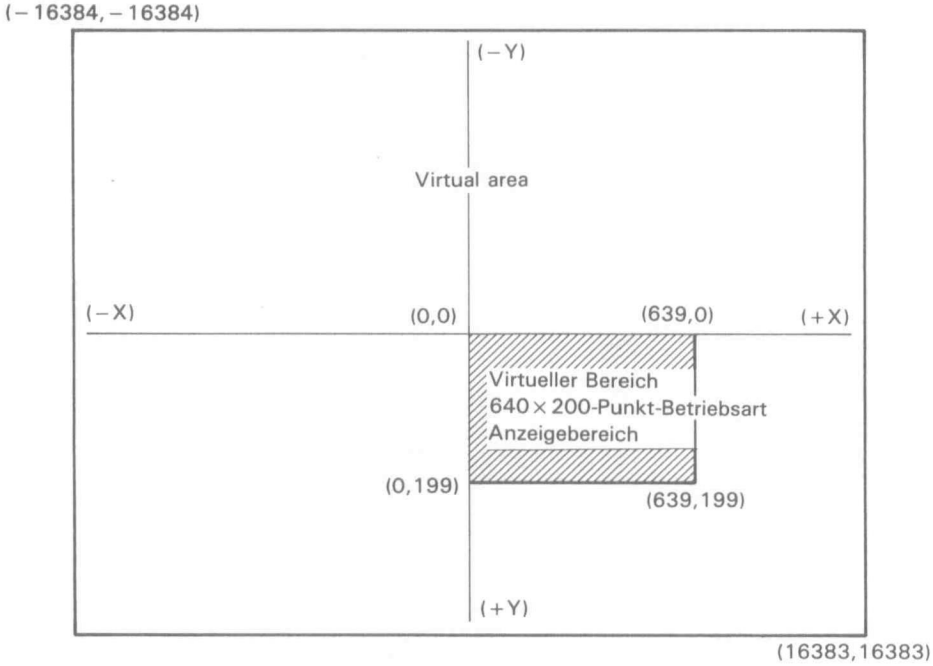
Format

SET [<Farbspezifikation>]<X-Koordinate>,<Y-Koordinate>
<Farbspezifikation> = [<Palettencode>][,<Betriebsart>]

Erklärung

Mit der SET-Anweisung wird an der durch <X-Koordinate> und <Y-Koordinate> spezifizierten Stelle in der spezifizierten Farbe ein Punkt auf dem Bildschirm gesetzt. <X-Koordinate> und <Y-Koordinate> sind numerische Ausdrücke (d.h. numerische Konstanten, Variablen oder Ausdrücke). Sie können Werte im folgenden Bereich annehmen:

$-16384 \leqq$ <X-Koordinate> $\leqq 16383$
$-16384 \leqq$ <Y-Koordinate> $\leqq 16383$



Obwohl man im virtuellen Bereich X -und Y-Koordinaten spezifizieren kann, zeigt BASIC nur den in der obigen Zeichnung schraffiert dargestellten Bereich an. Der <Palettencode> steht mit der Farbe des zu plottenden Punktes in Verbindung. Die <Betriebsart> muß entweder 0 oder 1 sein. Wenn 0 spezifiziert ist, wird der Punkt ungeachtet des gegenwärtigen Palettencodewertes in der durch <Palettencode> spezifizierten Farbe angezeigt. Wenn 1 spezifiziert ist, wird der Punkt in der durch den Farbcode spezifizierten Farbe angezeigt; um dies zu erhalten, bringt man den gegenwärtigen Palettencode mit dem in der SET-Anweisung spezifizierten <Palettencode> in eine ODER-Beziehung.
Wenn <Farbspezifikation> ausgelassen wird, so wird der Punkt in der durch die letzte COLOR-Anweisung spezifierten Farbe angezeigt.

Beispiel

10 SET[3,0] 100,50..... Bringt an den Koordinaten (100,50) einen Punkt in der mit Palettencode 3, Überlagerungs-Betriebsart 0 vorgegebenen Farbe zur Anzeige.

Siehe hierzu auch

RESET

RESET

Format

RESET [<Farbspezifikation>]<X-Koordinate>,<Y-Koordinate>
<Farbspezifikation> = <Palettencode>,<Überlagerungs-Betriebsart>

Erklärung

Die RESET-Anweisung verändert die Farbe eines auf dem Bildschirm durch <X-Koordinate> und <Y-Koordinate> spezifizierten Punktes gemäß der nachstehenden Regel. Die <X-Koordinate> und <Y-Koordinate> sind numerische Ausdrücke (d.h numerische Konstanten, Variable oder Ausdrücke). Sie können Werte im folgenden Bereich annehmen:

$-16384 \leqq$ <X-Koordinate> $\leqq 16383$
$-16384 \leqq$ <Y-Koordinate> $\leqq 16383$

Ihr Wertbereich ist der gleiche wie der für die SET-Anweisung. Der <Palettencode> spezifiziert den die rückzusetzende Farbe des Punktes betreffenden Palettencode. Die <Betriebsart> muß entweder 0 oder 1 sein. Für weitere Hinweise hierzu siehe Anhang A.

Siehe hierzu auch

SET

LINE

Format

LINE [<Farbspezifikation>]<X-Koordinate>,<Y-Koordinate>
<X-Koordinate>,<Y-Koordinate>[,<X-Koordinate>,<Y-Koordinate>]...
<Farbspezifikation> = [<Palettencode>][,<Betriebsart>

Erklärung

Die LINE-Anweisung zeichnet Linien, mit denen vorgegebene Punkte in der spezifizierten Farbe miteinander verbunden werden. Die <X-Koordinate> und <Y-Koordinate> sind numerische Ausdrücke (d.h. numerische Konstanten, Variablen oder Ausdrücke). Ihr Wertbereich ist der gleiche wie bei der SET-Anweisung. Der <Farbspezifikation>-Parameter ist mit dem der SET-Anweisung identisch. Wenn dieser Parameter ausgelassen wird,so wird die mit der COLOR-Anweisung vorgenommene Farbspezifikation vorausgesetzt. Wenn Koordinaten außerhalb des Anzeigebereiches spezifiziert werden, wird die Linie an der Grenze des Anzeigebereiches abgebrochen.

Beispiel

```
10 LINE [2,0]10,20,260,180,380,60
60 END
```

Das obige Programm zeichnet Linien zur Verbindung von den Punkten (10,20), (260,180) nach (380,60) und zwar in der Farbe von Palettencode 2 und der Überlagerungs-Betriebsart 0.

```
10  INIT "CRT:M1"
20  FOR X1=0 TO 319 STEP 3
30  LINE 159,99,X1,0
40  NEXT X1
50  FOR Y1=0 TO 199 STEP 3
60  LINE 159,99,319,Y1
70  NEXT Y1
80  FOR X2=319 TO 0 STEP -3
90  LINE 159,99,X2,199
100 NEXT X2
110 FOR Y2=199 TO 0 STEP -3
120 LINE 159,99,0,Y2
130 NEXT Y2
140 END
```

Das obige Programm zeichnet punktierte Linien (jeden dritten Punkt) von der Bildschirmmitte (159,99) zu den Ecken des Bildschirms.

Siehe hierzu auch

BLINE, SET

BLINE

Format

BLINE [<Farbspezifikation>]<X-Koordinate>,<Y-Koordinate>
<X-Koordinate>,<Y-Koordinate> [,<X-Koordinate>,<Y-Koordinate>]...
<Farbspezifikation> = <Palettencode>,<Überlagerungs-Betriebsart>

Erklärung

Die BLINE-Anweisung verändert die Farbe der Linien zur Verbindung auf dem Bildschirm vorgegebene Punkte gemäß der nachstend aufgeführten Regel.
<X-Koordinate> und <Y-Koordinate> sind numerische Ausdrücke (d.h numerische Konstanten, Variablen oder Ausdrücke). Ihr Wertbereich ist der gleiche wie bei der SET-Anweisung.
Der <Farbspezifikation> Parameter ist mit der RESET-Anweisung identisch. Wenn dieser Parameter ausgelassen wird, so wird die in der COLOR-Anweisung gemachte Farbspezifikation vorausgesetzt. Werden Koordinaten außerhalb des Anzeigebereiches spezifiziert, so wird nur das Liniensegment im Anzeigebereich gelöscht. Für weitere Hinweise siehe Anhang A.

Siehe hierzu auch

LINE, RESET

BOX

Format

BOX [<Farbspezifikation>]<X-Koordinate 1>,<Y-Koordinate 1>,
<X-Koordinate 2>,<Y-Koordinate 2>[,<Palettencode>]
<Farbspezifikation> = [<Palettencode>][,<Überlagerungs-Betriebsart>]

Erklärung

Die BOX-Anweisung zeichnet einen Kasten, wobei die 2 Koordinatenpaare als die Orte der gegenüberliegenden Ecken der Box verwendet werden. <X-Koordinate> und <Y-Koordinate> sind numerische Ausdrücke. Ihr Wertbereich ist der gleiche wie bei der SET-Anweisung.
Der <Farbspezifikation> Parameter ist mit der SET-Anweisung identisch. Wenn dieser Parameter ausgelassen wird, so wird die in der COLOR-Anweisung gemachte Farbspezifikation vorausgesetzt.
Der letzte <Palettencode> Parameter schreibt vor, daß der Kasten in der spezifizierten Farbe eingefärbt werden soll. Wenn dieser Parameter ausgelassen wird, so werden nur die Umrisse gezeichnet.

Beispiel

```
10 INIT''CRT:M1''
20 CLS
30 BOX [2,0]20,20,60,60,2
40 END
```

Dieses Programm zeichnet ein Rechteck auf dem Bildschirm und färbt es in der Farbe von Palette 2 ein.

Siehe hierzu auch

SET

CIRCLE

Format

CIRCLE [<Farbspezifikation>]<X-Koordinate>,<Y-Koordinate>,
<Radius>[,[<Aspekt>][,<Anfang>,<Ende>][,O]]
<Farbspezifikation> = [<Palettencode>][,<Überlagerungs-Betriebsart>]

Abgekürztes Format

CI.

Erklärung

Die CIRCLE-Anweisung zeichnet eine Ellipse (Kreis) bzw. einen Bogen (Fächer). Die Bedeutungen der <Farbspezifikation> Parameter sind mit denen der SET-Anweisung identisch. Wenn dieser Parameter ausgelassen wird, so wird der der COLOR-Anweisung vorausgesetzt.
<X-Koordinate> und <Y-Koordinate> geben die Koordinaten vom Mittelpunkt des Kreises an und <Radius> den Raduis des Kreises. Ihre Wertbereiche sind wie folgt:

$-19384 \leqq$ <Koordinaten> $\leqq 16383$
$0 \leqq$ <Radius> $\leqq 16383$

Der Bereich, in dem der Kreis effektiv angezeigt werden kann, wird durch den INIT-Befehl bestimmt.
<Aspekt> betrifft das Verhältnis von X-Radius zu Y-Radius. Wenn <Aspekt> kleiner als 1 ist, so ist der angegebene Radius der X-Radius. Wenn der <Aspekt> größer als 1 ist, so ist der Y-Radius angegeben. Der vorgegebene Standardwert von <Aspekt> ist 1.
Die Winkelparameter <Anfang> und <Ende> spezifizieren, wo das Zeichnen der Ellipse anfangen und enden soll. Diese Parameter müssen im Bogenmaß angegeben werden. Wenn man sie ausläßt, wird eine Ellipse (Kreis) gezeichnet. Wenn mit <Anfang> und <Ende> der O-Parameter spezifiziert wird, so wird ein Fächer gezeichnet, d.h. ein Kreisbogen, der durch mehrere Linien mit dem Mittelpunkt verbunden ist. Läßt man ihn aus, wird ein Kreisbogen gezeichnet.

Beispiel

```
10 INIT"CRT:M1"
20 CIRCLE[1,0]100,100,80,0.5
30 GOSUB 80
40 CIRCLE[2,0]50,130,60,0.5,0,π/4,O
50 GOSUB 80
60 CIRCLE 159,99,50
70 END
80 GET A$:IF A$=" " THEN 80
90 RETURN
```

Das obigen Programm zeichnet eine Ellipse auf dem Bildschirm dann wird, durch Drücken einer beliebigen Taste, ein Kreisbogen und ein Kreis gezeichnet.

Hinweis:
Im 640 × 200-Modus wird kein Kreis sondern eine Ellipse gezeichnet.

Siehe hierzu auch

SET, GET

PAINT

Format

PAINT [<Palettencode>]<X-Koordinate>,<Y-Koordinate>,
<Bereichsfarbe> [,<Bereichsfarbe>]...

Erklärung

Mit der PAINT-Anweisung wird ein bestimmter Bereich des Bildschirms mit der durch <Palettencode> spezifizierten Farbe ausgefüllt.
Wenn <Palettencode> ausgelassen wird, so wird die in der COLOR-Anweisung spezifizierte Palette vorausgesetzt. Man kann aus 16 Farben die <Bereichsfarbe> wählen. Der Bereich der Werte, die die <X-Koordinate> und <Y-Koordinate> annehmen können, wird durch den INIT-Befehl bestimmt.
Wenn der Bereich nicht vollständig durch die spezifizierte Umrißfarbe umschlossen ist (wir nennen diesen Zustand "geschlossener Kreis"), ragt die eingefärbte Fläche über diesen Bereich hinaus. Wenn die spezifizierten X- und Y-Koordinaten an den Grenzen oder innerhalb des bereits mit der spezifizierten Farbe gefärbten Bereiches liegen, wird der Färbevorgang unterdrückt.
Alle Zeichnungen werden in Form von Punkten erstellt, beim Zeichnen von Linien und Kurven in einem kleinen Bereich können daher kleine geschlossene Kreise entstehen. In diesem Falle wird das Färben nur ausgeführt, wenn innerhalb eines geschlossenen Kreises fallende Koordinaten spezifiziert werden.

Beispiel

```
10 INIT"CRT:M1"
20 CLS
30 CIRCLE[2,0]160,100,50
40 PAINT[1]160,100,2
50 END
```

Das obige Programm zeigt den durch eine Umrißlinie von Palettencode 2 umgebenen Bereich mit einer Farbe von Palettencode 1, wobei am Punkt (160,100) gestartet wird.

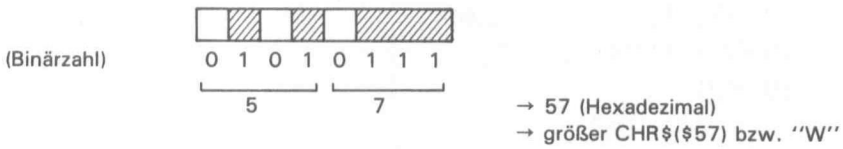
PATTERN

Format

PATTERN [<Farbspezifikation>]<numerische Daten>,<Textdaten>
<Farbspezifikation> = [<Palettencode>][,<Überlagerungs-Betriebsart>]

Erklärung

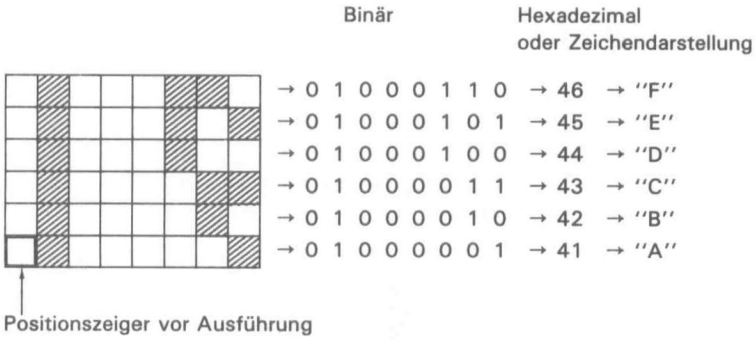
Die PATTERN-Anweisung definiert ein grafisches Muster in der spezifizierten Farbe. Die Bedeutung des <Farbspezifikation> Parameters ist mit der der SET-Anweisung identisch. Wenn dieser Parameter ausgelassen wird, so wird die COLOR-Anweisung vorausgesetzt.
Das zu zeichnende Muster kann unter Verwendung von <numerische Daten> und <Textdaten> spezifiziert werden. <Numerische Daten> (±1 bis ±24) dient zur Darstellung der Anzahl gestapelter 8-Bit Punktmusterreihen und die <Textdaten> verkörpern die einzelnen Punktmusterreihen.
Das Zeichnen des Musters wird durch den Positionszeiger gesteuert. Die durch <numerische Daten> spezifizierten Punktmusterreihen werden von unten nach oben angezeigt, wenn <numerische Daten> eine positive Zahl ist und von oben nach unten, wenn die Zahl negetiv ist. Nachdem die spezifizierte Anzahl Punktmusterreihen gezeichnet ist, werden die Punktmusterreihen um 8 Bit (1) nach rechts von der gegenwärtigen Spalte verschoben angezeigt. <Textdaten> müssen unter Verwendung des ASCII-Codes spezifiziert werden, der der Binärdarstellung von Punktmusterreihen entspricht.



Beispiel

```
10 POSITION 100,100........ Einrichtung des Positionszeigers
20 PATTERN[2,0]6,"ABCDEF"
30 END
```

Das obige Programm zeichnet das nachstehend gezeigte Grafikmuster in der Farbe von Palette 2.



Die obige Zeile 20 kann gegen folgende Zeile ausgetauscht werden.

```
20 PATTERN[2,0]6,CHR$($41,$42,$43,$44,$45,$46)
```

Siehe hierzu auch

POSITION

POSITION

Format

POSITION <X-Koordinate>,<Y-Koordinate>

Abgekürztes Format

POS.

Erklärung

Die POSITION-Anweisung setzt den Positionszeiger auf einen gegebenen Punkt auf dem Bildschirm. Der Positionszeiger zeigt zu der Position auf dem Bildschirm, an der das in der nachfolgenden PATTERN-Anweisung spezifizierte Punktmuster anzuzeigen ist. Der Wertbereich der <X-Koordinate> und <Y-Koordinate> Parameter ist der gleiche wie bei der SET-Anweisung.

Beispiel

```
10 POSITION 100,50
20 A$="ABCDEFGH"
30 PATTERN[1,0]-8,A$
40 END
```

Die POSITION-Anweisung in Zeile 10 setzt den Positionszeiger auf (100,50), an dieser Stelle beginnt die Ausführung der nachgestellten PATTERN-Anweisung.

Siehe hierzu auch

PATTERN

SYMBOL

Format

SYMBOL [<Farbspezifikation>]<X-Koordinate>,<Y-Koordinate>,
<Textdaten>[,<horizontale Vergrößerung>]*
[,<vertikale Vergrößerung>]*[,<Winkelcode>]
<Farbspezifikation> = [<Palettencode>][,<Überlagerungs-Betriebsart>]

Abgekürztes Format

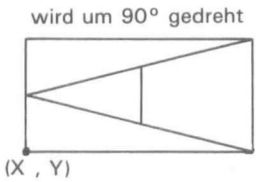
SY.

Erklärung

Die SYMBOL-Anweisung zeichnet ein Grafikmuster einer gegebenen Größe in einem gegebenen Winkel. Bei Auftreten dieser Anweisung positioniert BASIC die untere linke Ecke des Grafikmusters, dargestellt durch <Textdaten> am Punkt <X-Koordinate> und <Y-Koordinate>, und zwar um den <Winkelcode> gedreht und um den Faktor von <horizontale Vergrößerung> und/oder <vertikale Vergrößerung> vergrößert. Die Bedeutung des <Farbspezifikationparameters> ist mit der der SET-Anweisung identisch. Bei Auslassen dieses Parameters wird die COLOR-Anweisung vorausgesetzt.

Der Bereich der Werte, die die <X-Koordinate> und <Y-Koordinate> annehmen können, ist mit dem in der SET-Anweisung spezifizierten identisch. <Horizontale Vergrößerung> und <vertikale Vergrößerung> sind Ganzzahlen von 1 bis 255, die Standardvorgabe ist 1.
Wenn <Winkelcode> spezifiziert ist, so wird das Muster bezogen auf die linke Ecke des Musters am Punkt <X-Koordinate, Y-Koordinate> um den durch <Winkelcode> vorgegebenen Winkel gegen den Uhrzeigersinn gedreht. Die Bezugsposition des Musters bleibt nach der Drehung unverändert. Das Verhältnis zwischen Winkelcodes und Winkeln ist nachstehend aufgeführt.

Winkelcode	Winkel	
0	0°	(Standardvorgabe)
1	90°	
2	180°	
3	270°	



Beispiel

```
10 SYMBOL [1] 40,0,''MZ–800'',5,5,0
20 FOR J=1 TO 3
30 SYMBOL [J] J*80,100,''A'',J+2,J+2,J
40 NEXT J
50 SYMBOL [1] 280,199,''MZ–800'',5,5,2
```

POINT

Format

POINT [<X-Koordinate>,<Y-Koordinate>

Erklärung

Die POINT-Funktion gibt den Palettencode an, der an der gegebenen Stelle auf dem Bildschirm definiert ist. Der Bereich der Werte, den <X-Koordinate> und <Y-Koordinate> annehmen können, wird durch den INIT-Befehl festgelegt.

Beispiel

```
10 INIT''CRT:M1''
20 SET[3,0]100,100
30 A=POINT(100,100)
40 PRINT A
50 END
RUN
3
Ready
```

Die Anweisung in Zeile 10 weist dem Palettencode 3 den Punkt (100,100) zu. Die POINT-Funktion gibt daher einen Palettencode von 3 an.

6.5 Musiksteuer-Befehle

MUSIC

Format

MUSIC <Ton 1 von Melodie 1>[;<Ton 1 von Melodie 2>]
[;<Ton 1 von Melodie 3>][,<Ton 2 von Melodie 1>]
[;<Ton 2 von Melodie 2>][;<Ton 2 von Melodie 3>]...

Abgekürztes Format

MU.

Erklärung

Die MUSIC-Anweisung generiert die durch die Zeichenfolge oder Stringvariable des Arguments spezifizierte Melodie bzw. Klangeffekt über Lautsprecher.
Drei Melodieteile können gleichzeitig gespielt werden. In der auf das Schlüsselwort MUSIC folgenden Musikspezifikation werden Semikolons zur Trennung der einzelnen Teile verwendet; Kommas dienen zur Trennung der einzelnen Melodien voneinander, wobei eine Melodie eine Folge von Tönen ist. Die Spezifikation aller Töne ist wie folgt:

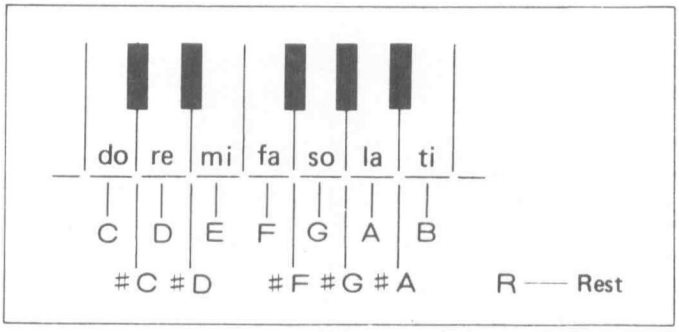
<Oktavenbestimmung> <Notenname> <Dauer>

(i) Oktavenbestimmung
Die Grundoktave einer Melodie wird im Format "On" spezifiziert, wobei n eine Zahl im Bereich 0 bis 6 ist. Das Pluszeichen dient dazu, die folgenden Töne eine Oktave höher als die Grundoktave zu spielen, wohingegen das Minuszeichen die umgekehrte Wirkung ausübt. Wenn kein Vorzeichen spezifiziert wird, so wird die Grundoktave vorausgesetzt. In einer Melodie, in der als Grundoktave O6 oder OO vorgegeben ist, ist die Verwendung von Pluszeichen bzw. Minuszeichen nicht gestattet.

(ii) Tonvorgabe
Zur Vorgabe von Noten innerhalb einer Oktave werden die folgenden Symbole verwendet:

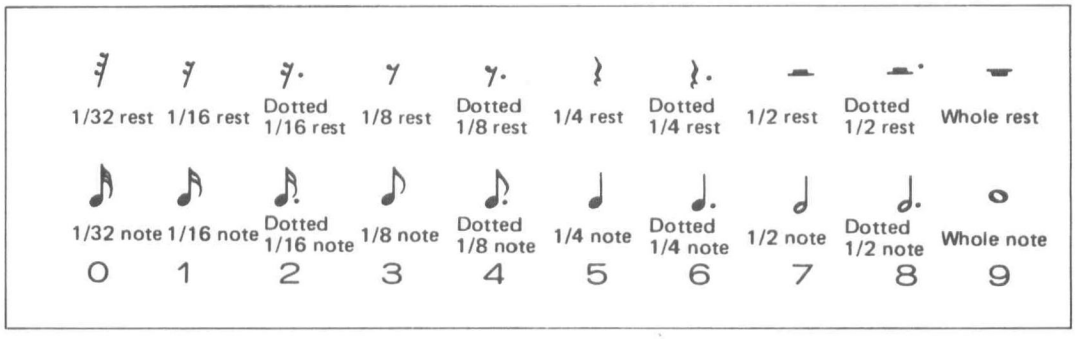
CDEFGAB#R

Das Verhältnis zwischen der 8-Noten-Tonleiter (do, re, mi, fa, so , la, ti, do) und diesen Symbolen ist nachträglich aufgeführt. Zur Erhöhung eines Tones um einen halben Ton verwenden wir das Symbol (#). Zum Senken um einen halben Ton versehen wir den nächstniedrigeren Ton mit einem (#)-Symbol. So wird z.B. b durch #A dargestellt. Pausen werden durch "R"angewiesen.



(iii) Tondauer-Vorgabe

Mit der Tondauer-Vorgabe wird bestimmt, wie lange der vorgegebene Ton gehalten werden soll. Wie nachstehend gezeigt, werden die Notenwerte von 1/32 bis zu ganzen Noten als Zahlen von 0 bis 9 spezifiziert. (Wenn R vorgegeben ist, wird durch diese Angabe die Länge der Pause bestimmt).



In der <Melodie> Spezifikation können die folgenden Optionen bestimmt werden:

	n = Ganzzahlwert	Standardvorgabe wert	Beschreibung
On	0 bis 6	2	Bestimmt die gegenwärtige Oktave. Wenn n = 2, beträgt die Frequenz von A 440 Hz.
Nn	0 bis 83		Spezifiziert den Ton direkt durch die entsprechende Notennummer anstatt durch die Oktavennummer und den Namen der Note. Die Werte von n sind in der Tabelle auf der folgenden Seite aufgelistet. N0 bis N8 bedeutet Pause.
Tn	1 bis 7	4	Zur Einstellung des Tempos in der gleichen Weise wie mit der TEMPO-Anweisung.
Vn	0 bis 15	15	Stellt die Lautstärke ein. Wenn n = 15, beträgt die Lautstärke Maximum, wenn n = 0, wird kein Ton generiert.
Sn	0 bis 7	8	Zur Vorgabe des Hüllkurven-Musters (Klang-Wellenform). Bezüglich der Werte von n siehe Abbildung der Hüllkurven auf der folgenden Seite.
Mn	1 bis 255 1 = ca. 10m/s	255	Wird stets mit dem S-Parameter verwendet, um die Geschwindigkeit zu spezifizieren, mit der die Hüllkurve sich verändern soll. Wenn n = 1, ist die Geschwindigkeit Maximum, mit Zunahme des Wertes von n nimmt die Geschwindigkeit ab und die Steilheit der Hüllkurve wird geringer. Zu große Mn-Werte erzeugen, je nach Hüllkurvenmuster möglicherweise unhörbare Töne.
Ln	0 bis 9	5	Bestimmt die Länge des Tons

Beispiel

Das folgende Programm spielt die Melodie "Oh, Susanna", eine Komposition von Stephen Foster.

```
10 TEMPO 6
20 A1$ = "O2G1A1B3 + D3 + D3 + E3 + D3B3G4A1B3B3A3G3A6G1A1B3 + D3 +
   D3 + E3 + D3B3G4A1B3B3A3A3G6R3"
30 A2$ = "G1A1G3B3B3 + C3B3G3G4G1G3G3G3G3E6G1A1G3B3B3 +
   C3B3G3G4G1G3G3E3E3G6R3"
40 MUSIC A1$;A2$
50 END
```

Tonnummern (Nn)

Ton \ Oktave	0	1	2	3	4	5	6
do		12	24	36	48	60	72
do #		13	25	37	49	61	73
re		14	26	38	50	62	74
re #		15	27	39	51	63	75
mi		16	28	40	52	64	76
fa		17	29	41	53	65	77
fa #		18	30	42	54	66	78
so		19	31	43	55	67	79
so #		20	32	44	56	68	80
la	9	21	33	45	57	69	81
la #	10	22	34	46	58	70	82
ti	11	23	35	47	59	71	83

Hüllkurvenfeld (Sn)

n	Hüllkurvenfeld (Die X-Achse steht für Zeit, die Y-Achse für Lautstärke).
0	
1	
2	
3	
4	
5	
6	
7	
8	

Die X-Achse dient zur Darstellung der Zeit in der mit Mn spezifizierten Einheit.

Die MUSIC-Anweisung bewirkt das Zwischenspeichern der Musikdaten und die Erzeugung von Tönen unabhängig von der Datenverarbeitung des Computers. Dadurch wird z.B. ermöglicht, daß während der Wiedergabe von Musik die Bildschirmanzeige verändert wird. Dies bedeutet aber auch, daß es unmöglich ist, die Musik mit der [SHIFT] -und [BREAK] -Taste zu drücken oder die Musikwiedergabe auf die Anzeige abgestimmt auszuführen. Um den Start- und Stopbetrieb der Musikwiedergabe zu steuern , sind die folgenden Befehle vorgesehen:

MUSIC STOP: Stoppt die Tonerzeugung

MUSIC WAIT: Unterbricht die Programmausführung, bis die gesamte Serie der Töne abgespielt ist.

MUSIC INIT: Initialisiert (startet) die Musik und stellt den Rauschpegel auf "O2V15L5T4S8M255" ein. (Siehe Tabelle auf der vorherigen Seite).

TEMPO

Format

TEMPO <numerischer Ausdruck>

Abgekürztes Format

TE.

Erklärung

Die TEMPO-Anweisung bestimmt das Tempo, mit dem die Musik durch die MUSIC-Anweisung gespielt wird. Das Tempo kann im Bereich von 1 bis 7 eingestellt werden. Standardvorgabe ist 4.

TEMPO 1: langsamstes Tempo
TEMPO 4: mittleres Tempo
TEMPO 7: schnellstes Tempo

SOUND

Format

SOUND <Tonhöhe>,<Dauer>
SOUND=(<Register>,<Daten>)

Abgekürztes Format

SO.

Erklärung

Die SOUND-Anweisung erzeugt Töne gemäß der Spezifikation durch <Tonhöhe> und <Dauer>.

(i) <Tonhöhe>
Mit Tonhöhe wird die Tonhöhe spezifiziert. Die Tonhöhe-Codes und die zugehörigen Noten sind in der nachstehenden Tabelle aufgeführt.

Note \ Oktave	0	1	2	3	4	5	6
do		12	24	36	48	60	72
do #		13	25	37	49	61	73
re		14	26	38	50	62	74
re #		15	27	39	51	63	75
mi		16	28	40	52	64	76
fa		17	29	41	53	65	77
fa #		18	30	42	54	66	78
so		19	31	43	55	67	79
so #		20	32	44	56	68	80
la	9	21	33	45	57	69	81
la #	10	22	34	46	58	70	82
ti	11	23	35	47	59	71	83

Die Frequenz der Note la in Oktave 2 ist 440 Hz.

(ii) <Dauer>

<Dauer> spezifiziert die Länge des durch diese Anweisung erzeugten Tons in 1/100 sec. <Dauer> muß ein numerischer Ausdruck 0 bis 65535 sein.
Das Format SOUND = (<Register>, <Daten>) dient dazu, den Tongenerator-LSI (PSG = Programmable Sound Generator) direkt zu steuern. Der PSG kann drei Töne und ein Rauschen erzeugen. Die PSG-Registertabelle ist nachstehend aufgeführt.

Register-nummer			Daten
0	Ton 0	Frequenz	Ganzzahl von 1 bis $2^{10} - 1$
1	Ton 0	Lautstärke	Ganzzahl von 0 bis 15 (siehe Hinweis) 9.
2	Ton 1	Frequenz	gleich wie Register 0
3	Ton 1	Lautstärke	gleich wie Register 1
4	Ton 2	Frequenz	gleich wie Register 0
5	Ton 2	Lautstärke	gleich wie Register 1
6	Rauschen	Frequenz	Rauschdaten
7	Rauschen	Lautstärke	gleich wie Register 1

Der PSG kann Synchronrauschen oder weißes Rauschen erzeugen. Welche Art von Rauschen erzeugt werden soll, wird durch Übertragung der 1-Byte-Daten am PSG-Anschluß $F2 spezifiziert.
Zur Steuerung des PSG siehe Anhang B.

NOISE

Format

NOISE <Melodie>[,<Melodie>] ...

$$<\text{Melodie}> = \begin{Bmatrix} + \\ - \end{Bmatrix} <\text{Notenname}> [<\text{Dauer}>]$$

Abgekürztes Format

NO.

Erklärung

Die NOISE-Anweisung erzeugt, wie durch die <Melodie n> Parameter spezifiziert, weißes Rauschen. Die Bedeutung der <Melodie n> Parameter ist mit denen der MUSIC-Anweisung identisch.
Diese Anweisung kann zwei Rauschteile gleichzeitig erzeugen. Die Parameter zum Spezifizieren dieser zwei Teile müssen durch ein Semikolon (;) voneinander getrennt werden. Je zwei aufeinanderfolgende Melodien müssen durch ein Komma (Komma) voneinander getrennt werden.

Beispiele

```
10 NOISE"C3D1","E3F1"
```

Die obige NOISE-Anweisung erzeugt zwei Teile von weißem Rauschen gleichzeitig.

6.6 Druckersteuer-Befehle

PTEST

Format **PTEST**

Abgekürztes Format

PTE.

Erklärung Der PTEST-Befehl bewirkt Ausdrucken von Quadraten in schwarz, blau, grün und rot in dieser Reihenfolge, um auf diese Weise die Farbspezifikation, Menge der Tinte in den Stiften usw. zu überprüfen.

0	1	2	3	← mit PCOLOR spezifizierter Wert
(Schwarz)	(Blau)	(Grün)	(Rot)	

Beispiel PTEST ..... überprüft den Stift.

PMODE

Format **PMODE** {TN, TL, TS, GR}

Abgekürztes Format

PM.

Erklärung Der PMODE-Befehl spezifiziert die Betriebsart für den Farbplotter.

PMODE TN

Der PMODE TN-Befehl dient zur Umschaltung des Druckers von der Grafik-Betriebsart zurück auf die Text-Betriebsart und gibt als Zeichengröße 40 Zeichen pro Zeile vor. (Standardvorgabe)

PMODE TL

Der PMODE TL-Befehl bewirkt die Umschaltung des Druckers von der Grafik-Betriebsart zurück auf Text-Betriebsart und gibt als Zeichengröße 26 Zeilen pro Zeile vor.

PMODE TS

Der PMODE TS-Befehl bewirkt die Umschaltung des Druckers von der Grafik-Betriebsart zurück auf die Text-Betriebsart und gibt als Zeichengröße 80 Zeichen pro Zeile vor.

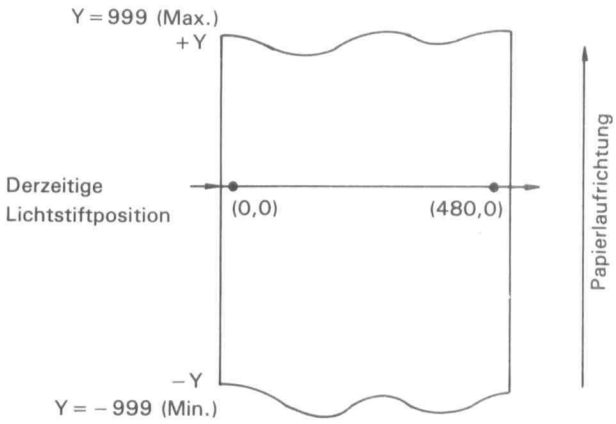
PMODE GR

Mit dem PMODE GR Befehl wird der Drucker von der Text-Betriebsart auf die Grafik-Betriebsart umgeschaltet. Zur Umschaltung auf diese Betriebsart sollte im gerade ausgeführten BASIC-Programm an der Stelle vor der Betriebarten-Umschaltung eine Eintragung der zur Zeit verwendeten Zeichengröße vorgenommen werden. Dies gestattet dem Programm dann die Rückkehr in die Text-Betriebsart, wenn die Tasten [SHIFT] + [BREAK] gedrückt werden oder ein STOP-Befehl auftritt.

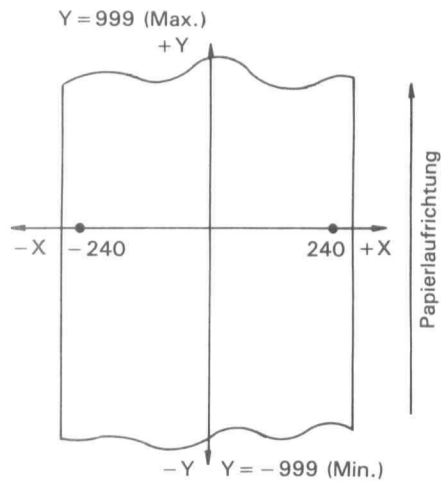
* * * Zeichen-Betriebsart * * *

SHARP MZ-800	80-Zeichen-Betriebsart (~~TL~~) TS
SHARP MZ-800	40-Zeichen-Betriebsart (TN)
SHARP MZ-800	26-Zeichen-Betriebsart (TS) TL

Mit dem PMODE GR-Befehl wird jeder in der Grafik-Betriebsart ausführbare Befehl aktiviert und die X- und Y-Achsen des Druckers werden vorgegeben. Die gegenwärtige Stiftposition wird als Ursprung (0,0) festgelegt. Der Ursprung kann zu jeder beliebigen Stelle verschoben werden, vorausgesetzt er befindet sich innerhalb des druckbaren Bereiches. Durch Ausführung des Druckbetriebes außerhalb der vorgegebenen Grenzen können Stifte beschädigt und Betriebsstörungen verursacht werden.



X- und Y-Achsen nach Ausführung eines PMODE GR-Befehls. Die X-Achse wird von 0 bis 480 gezeichnet und die Y-Achse von −999 bis 999.



X- und Y-Achse nach Verschiebung des Ursprunges zum Mittelpunkt (PMOVE 240, −240: HSET)

PCOLOR

Format

PCOLOR $\left\{\begin{matrix}0\\1\\2\\3\end{matrix}\right\}$

Abgekürztes Format

PC.

Erklärung

Der PCOLOR-Befehl spezifiziert die zum Ausdrucken von Zeichen oder Grafik verwendete Farbe. Dieser Befehl kann in der Text-Betriebsart oder der Grafik-Betriebsart eingegeben werden. Die verfügbaren Farben und die entsprechenden Farbennummern sind nachstehend aufgeführt.

0: Schwarz
1: Blau
2: Grün
3: Rot

Beispiel

10 PCOLOR 1 ..... Bestimmt als Stiftfarbe blau.

PSKIP

Format

PSKIP <Anzahl der Zeilen>

Abgekürztes Format

PS.

Erklärung

Der PSKIP-Befehl bewirkt ab hier Vorschub bis zur spezifizierten <Anzahl der Zeilen> in Vorwärtsrichtung, wenn der spezifizierte Wert positiv ist und er bewegt das Papier um die spezifizierte <Anzahl der Zeilen> rückwärts, wenn der Wert negativ ist. <Anzahl der Zeilen> muß eine Ganzzahl von −20 bis 20 sein. Dieser Befehl gilt nur in der Text-Betriebsart.

Beispiel

10 PSKIP 12..... Bewirkt Papiervorschub um 12 Zeilen vorwärts.
20 PSKIP −6..... Bewirkt Papiervorschub um 6 Zeilen rückwärts.

PAGE

Format

PAGE <Anzahl der Zeilen>

Abgekürztes Format

PA.

Erklärung

Der PAGE-Befehl spezifiziert die Anzahl der Zeilen pro Seite. (Anzahl der Zeilen) muß eine Ganzzahl von 1 bis 72 sein. Desgleichen wird mit dem PAGE-Befehl die gegenwärtige Seitenposition als die erste Zeile der Seite vorgegeben. Dieser Befehl kann nur in der Text-Betriebsart ausgeführt werden.

Beispiel

10 PAGE 20 ..... Legt als Anzahl der Zeilen pro Seite 20 fest. Mit dieser Einstellung wird bei Ausführung eines Formularvorschubs eine Seite jeweils als 20 Zeilen gehandhabt.

PRINT/P

Format

PRINT/P <Daten>[{ , ; }<Daten>]...

Abgekürztes Format

?/P

Erklärung

Der PRINT/P-Befehl bewirkt die Ausgabe von Daten an den Drucker fast im gleichen Format wie die Ausgabe von Daten mit dem PRINT-Befehl auf dem Bildschirm. Sowohl die Separatoren "," und ";" als auch die TAB-Funktion in diesem Befehl haben die gleiche Wirkung wie beim PRINT-Befehl. Verschiedene vom Drucker unterstützte Funktionen können effizient verwendet werden, und zwar indem Druckersteuercodes im folgenden Format übertragen werden:

PRINT/P CHR $ (Steuercode)

Beispiele

10 PRINT/P "ABCD"..... druckt "ABCD".
10 PRINT/P CHR$ ($OA)..... bewirkt einen Zeilenvorschub.

Hinweis:
Zur Ausführung von PRINT/P-Anweisungen, die den Steuercode ↑ oder ↓ nacheinander enthalten, empfiehlt es sich, diese auf getrennten Zeilen zu schreiben oder sie durch (:) zu begrenzen. Anderenfalls interpretiert BASIC die durch Verknüpfungselemente (+) verbundenen Steuercodes als eine einzige Codefolge, wodurch eine Betriebsstörung des Druckers verursacht werden kann.

[Ungültig] PRINT/P" ↓ "+" ↓ ↓ 4D"

BASIC interpretiert diese Anweisung als PRINT/P " ↓ ↓ ↓ "+"4D". Stattdessen schreibt man den Druckersteuercode im folgenden Format:

[Gültig] PRINT/P" ↓ ":PRINT" ↓ ↓ 4D"

PRINT/P USING

Format

PRINT/P [<Palettencode>]**USING**"Formatstring";<Daten>[{;,} <Daten>]...

Abgekürztes Format

?/P USI.

Erklärung

Gleich wie die PRINT USING-Anweisung, abgesehen davon, daß das Ausgabegerät der Drucker ist.

Siehe hierzu auch

PRINT USING

PLINE

Format

PLINE [% <Linientyp>,] x1,y1 [,x2,y2,···,xi,yi]

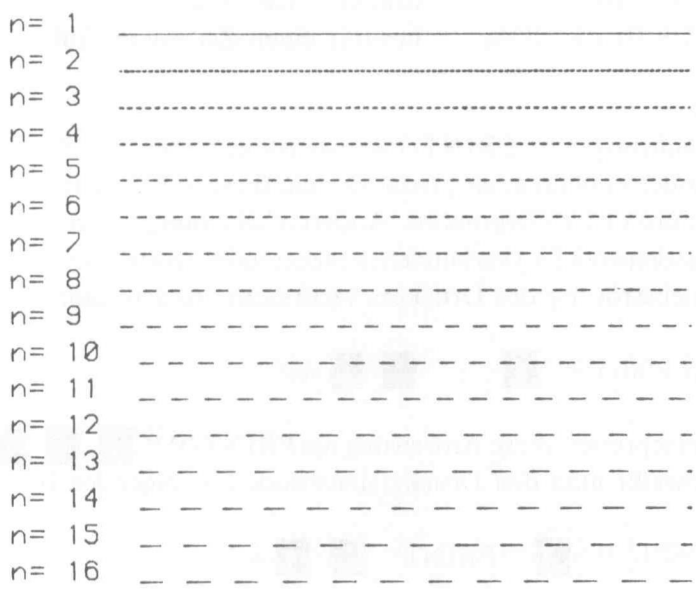
Abgekürztes Format

PLI.

Erklärung

Mit dem PLINE-Befehl wird eine durchgehende oder eine punktierte Linie von der gegenwärtigen Stiftposition bis zu der durch die absoluten Koordinaten (x1,y1) angezeigten Position gezogen, daraufhin wird eine Linie von dieser Stelle bis zu der durch die absoluten Koordinaten (x2,y2) ausgewiesenen Position gezogen usw. xi muß eine Ganzzahl von −480 bis 480 sein und yi eine Ganzzahl von −999 bis +999. <Linientyp> spezifiziert den Typ der zu ziehenden Linie und muß eine Ganzzahl von 1 bis 16 sein. Durchgehende Linien werden gezogen, wenn <Linientyp> = 1 und punktierte Linien, wenn <Linientyp> = 2 bis 16 beträgt, wobei die Zahl n jeweils einem Linientyp entspricht. Wenn % <Linientyp> ausgelassen wird, so wird der vorherige Wert von n vorausgesetzt. Der ursprüngliche Wert von <Linientyp> ist 1 (durchgehende Linie). Die durch n ausgewiesene Linie ist wie folgt:

* * * ZEILE 1—16 * * *



Der PLINE-Befehl ist nur in der Grafik-Betriebsart gültig.

Beispiel

```
10 PMODE GR
20 PLINE %1, 0, 0, 200, 0, 200, -200, 0, -200, 0, 0
30 END
```

Das obige Programm zeichnet ein Quadrat mit einer Seitenlänge von 200 Einheiten.

RLINE

Format

RLINE %[<Linientyp>,] x1,y1 [,x2,y2,···,xi,yi]

Erklärung

Der RLINE-Befehl zeichnet eine Linie von der gegenwärtigen Stiftposition bis zur der durch die relativen Koordinaten (x1,y1) angezeigten Position, daraufhin zeichnet er eine Linie von dieser Stelle bis zu der durch die relativen Koordinaten (x2,y2) angezeigten Positionen usw.
(xi) muß eine Ganzzahl von −480 bis 480 sein und (yi) muß eine Ganzzahl von −999 bis 999 sein. Die Spezifikationen für <Linientyp> sind die gleichen wie mit dem PLINE-Befehl. Der RLINE-Befehl gilt nur in der Grafik-Betriebsart.

Beispiel

```
10 PMODE GR
20 SQ=INT (120*SQR(3))
30 RLINE %1,240,0,-120,-SQ,-120,SQ
40 PMODE TN
```

Dieses Programm zeichnet ein Dreieck mit durchgehenden Linien.

PMOVE

Format

PMOVE <X-Koordinate>,<Y-Koordinate>

Erklärung

Der PMOVE-Befehl hebt den Stift und bewegt ihn zur spezifizierten Position (x,y). <X-Koordinate> und <Y-Koordinate> müssen Ganzzahlen im Bereich von −480 bis 480 bzw. −999 bis 999 sein. Dieser Befehl gilt nur in der Grafik-Betriebsart.

Beispiel

Das folgende Programm zeichnet ein Kreuz mit einer Seitenlänge von 480 Einheiten:

```
10 PMODE GR
20 PLINE 0,0,480,0
30 PMOVE 240,240
40 PLINE 240,240,240,-240
50 PMODE TN
```

Vor Ausführung dieses Programms auf jeden Fall das Papier entsprechend weit vorbewegen.

RMOVE

Format **RMOVE** <X-Koordinate>,<Y-Koordinate>

Abgekürztes Format

RM.

Erklärung Der RMOVE-Befehl hebt den Stift und bewegt ihn zu der durch die relativen Koordinaten (x,y) angezeigten Position. <X-Koordinate> muß eine Ganzzahl von −480 bis 480 sein und <Y-Koordinate> eine Ganzzahl von −999 bis 999. Der RMOVE-Befehl gilt nur in der Grafik-Betriebsart.

Beispiel

```
10 PMODE GR
20 PMOVE 240,0
30 PLINE 240,0,360,120
40 RMOVE −120,0
50 PLINE 240,120,360,240
60 PMODE TN
```

Das obige Programm zeichnet zwei schräge Linien.
Vor Ausführung dieses Programms auf jeden Fall das Papier entsprechend weit vorbewegen.

PHOME

Format **PHOME**

Abgekürztes Format

PH.

Erklärung Der Befehl führt den Stift in Ausgangsposition zurück.
Dieser Befehl gilt nur in der Grafik-Betriebsart.

Beispiel

```
10 PMODE GR
20 PLINE 240,−240
30 PCIRCLE 240,−240,50
40 PHOME ..................... Führt den Stift in Ausgangsposition zurück.
50 PMODE TN
```

HSET

Format **HSET**

Abgekürztes Format

H.

Erklärung

Der HSET-Befehl bestimmt die gegenwärtige Stiftposition als den neuen Ursprung. Bevor man einen HSET-Befehl anweist, empfiehlt es sich, mit dem PMOVE-Befehl den Stift zu der Stelle zu bewegen, die am besten zum Zeichnen des Gewünschten geeignet ist. Dieser Befehl gilt nur in der Grafik-Betriebsart.

Beispiel

```
10 PMODE GR
20 PMOVE 240,-240
30 HSET ........................ Gibt (240,-240) als den neuen Ursprung (0,0) vor.
40 PMOVE 240,0
50 PLINE 240,0,0,-240,-240,0,0,240,240,0
60 PHOME
70 PMODE TN
```

GPRINT

Format **GPRINT** [[<Größe>,<Winkel>],]<Textdaten>

Abgekürztes Format

GP.

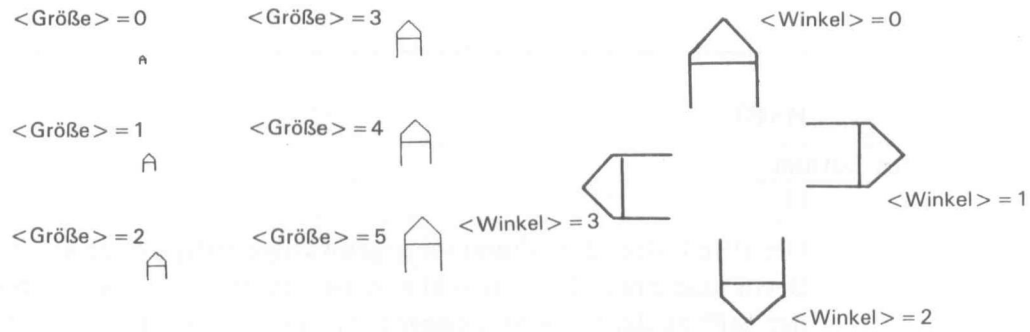
Erklärung

Der GPRINT-Befehl dient zum Drucken des spezifizierten Zeichens mit spezifizierter Größe und Winkel.
<Größe> kann eine beliebige Zahl von 0 bis 63 sein. 80 Zeichen pro Zeile können ausgedruckt werden, wenn <Größe> = 0; 40 Zeichen pro Zeile wenn <Größe> = 1; und 26 Zeichen pro Zeile wenn <Größe> = 2. <Winkel> zeigt die Richtung an, in der die Zeilen gedruckt werden. <Winkel> muß eine Ganzzahl von 0 bis 3 sein. Die Rotation des Zeichens wird bezogen auf die linke untere Ecke ausgeführt. Wenn <Größe> und <Winkel> ausgelassen werden, so werden die vorherigen Einstellungen vorausgesetzt. Die Ausgangswerte (Standardvorgabe) für <Größe> = 1 und <Winkel> = 0.
Der GPRINT-Befehl gilt nur in der Grafik-Betriebsart.

Beispiel

```
10 PMODE GR
20 GPRINT "A" ............... Druckt "A" in der Grafik-Betriebsart.
30 PMOVE 240,-240
40 GPRINT[2,2],"A"
50 PHOME
60 PMODE TN
```

Die folgenden Abbildungen zeigen verschiedene Druckbeispiele.



AXIS

Format **AXIS** <Achse>,<Punktabstand>,<Wiederholungen>

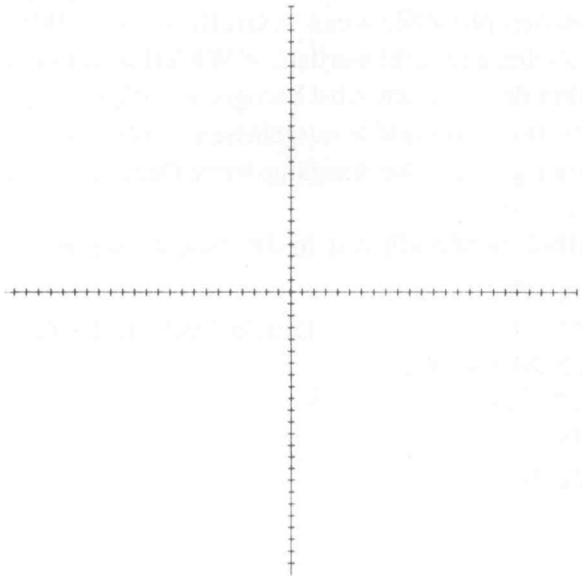
Abgekürztes Format

AX.

Erklärung Der AXIS-Befehl zeichnet die X-Achse wenn <Achse> = 1 und die <Y-Achse> wenn <Achse> = 0. Die Anzahl der in <Wiederholungen> vorgegebenen Skalenmarkierungen wird mit dem durch <Punktabstand> markierten Punktabstand gezeichnet.
<Punktabstand> muß eine Ganzzahl von −999 bis 999 sein.
<Wiederholungen> muß eine Ganzzahl von 1 bis 255 sein.
Der AXIS-Befehl gilt nur in der Grafik-Betriebsart.

Beispiel

```
10 PMODE GR
20 PMOVE 240,0
30 AXIS 0,-10,48
40 PMOVE 0,-240
50 AXIS 1,10,48
60 PMODE TN
```



Das obige Beispielprogramm zeichnet die X -und Y-Achsen mit Skalenmarkierungen von −240 bis 240 in Abständen von 10er-Einheiten.
Nach Festlegung des Schnittpunktes der X- und Y-Achse als neuer Ursprung, können die Koordinaten in der gleichen Weise verwendet werden wie gewöhnliche (rechtwinklige, kartesische) Koordinaten. (X = −240 bis 240, Y = −240 bis 240).

PCIRCLE

Format

PCIRCLE <X-Koordinate>,<Y-Koordinate>,<Radius>,
<Startwinkel>,<Endwinkel>,<Schrittwinkel>.
<X-Koordinate>: −999 bis 999
<Y-Koordinate>: −999 bis 999
<Radius>: 0 bis 999

Abgekürztes Format

PCI.

Erklärung

Der PCIRCLE-Befehl zeichnet einen Kreis oder einen Kreisbogen im Gegenuhrzeigersinn mit dem <Radius> und einem <Schrittwinkel> mit dem Mittelpunkt an der Position (x,y), angefangen am <Startwinkel> und endend am <Endwinkel>. Wenn <Startwinkel> = 0, <Endwinkel> = 360 und <Schrittwinkel> = 0,2 wird ein vollständiger Kreis gezeichnet. Tatsächlich zeichnet dieser Befehl ein Polygon; <Schrittwinkel> muß daher so klein wie möglich sein, um eine glatte Linienführung zu verwirklichen. <Startwinkel> muß kleiner als <Endwinkel> sein. Wenn <Schrittwinkel> = 0 werden Linien zur Verbindung von Mittelpunkt und Startpunkt, desgleichen Mittelpunkt und Endpunkt gezeichnet. Der PCIRCLE-Befehl gilt nur in der Grafik-Betriebsart.

Beispiel

```
10 PMODE GR:P=0
20 PMOVE 240,-240
30 HSET
40 FOR I=240 TO 40 STEP -60
50 PCOLOR P
60 PCIRCLE 0, 0, I, 0, 360, 2
70 P=P+1
80 NEXT I
90 PMODE TN
```

LIST/P

Format **LIST/P** [<Anfangszeilenummer>][–][<Endzeilenummer>]

Abgekürztes Format

L./P

Erklärung Der LIST/P-Befehl dient zum Auflisten aller im Speicher stehenden Programmzeilen mit dem Drucker. Zum Auflisten des gesamten Programms ist die Angabe eines Operanden nicht erforderlich. Um nur einen Teil der Programmzeilen aufzulisten, gibt man die <Anfangszeilennummer> oder die <Endzeilennummer> oder beide mit einem Bindestrich verbunden an. Dieser Befehl kann nur in der Text-Betriebsart ausgeführt werden.

HCOPY

Format **HCOPY**

Abgekürztes Format

HC.

Erklärung Der HCOPY-Befehl kopiert den Bildschirminhalt auf den Drucker. Dieser Befehl steht nur für den Drucker MZ-80P5(K) zur Verfügung, er kann nicht mit dem Farbplotter verwendet werden.

PLOT

Format **PLOT** { ON / OFF }

Abgekürztes Format

PL.

Erklärung

Der PLOT ON-Befehl ermöglicht die Verwendung des Farbplotters als Anzeigegerät. Mit anderen Worten kann man den MZ-800 so auch ohne zusätzliche Bildschirmanzeige verwenden. Mit dem PLOT ON-Befehl wird die Anzahl der pro Zeile gedruckten Zeichen auf 80 eingestellt, wenn die 80-Spalten-Betriebsart für den Bildschirm aktiviert ist und auf 40 Zeichen, wenn die 40-Spalten-Betriebsart für den Bildschirm aktiviert ist.
Dieser Befehl gilt nur, wenn der Farbplotter installiert und in der Text-Betriebsart verwendet wird. Durch Ausführung eines PLOT ON-Befehls wird der CONSOLE-Befehl ungültig. Alle Zeichen, die nicht im Zeichengenerator des Farbplotters enthalten sind, werden durch ein "." dargestellt. Durch Ausführung dieses Befehls werden die Tasten INST, DEL und [←] außer Betrieb gesetzt. Um die Stiftfarbe zu verändern, kann man [CTRL] + [G] verwenden.
Der PLOT OFF-Befehl löscht den PLOT ON-Befehl.
Desgleichen wird der PLOT ON-Befehl durch INIT "CRT:Mn" gelöscht.
Der Plotter-Drucker ist auf 40-Zeichen-Betrieb eingestellt, wenn die PLOT ON Anweisung ausgeführt wird, während die Anzeige auf 40 Zeichen eingestellt ist. Wenn die Anzeige auf 80 Zeichen eingestellt ist, wird, er auf 80-Zeichen Betrieb eingestellt.

• Drucker-Betriebsarten

In der nachstehenden Tabelle sind die Betriebsarten, Betriebsartenwahl-Befehle und die verwendbaren Befehle aufgeführt.

Betriebsart	Betriebsartenwahl-Befehl	Verwendb. Befehle	
Text-Betriebsart 40 Zeichen/Zeilen	PMODE TN	PTEST PCOLOR PSKIP PAGE * PRINT/P * PRINT/P USING	* LIST/P * HCOPY PLOT
Text-Betriebsart 26 Zeichen/Zeilen	PMODE TL		
Text-Betriebsart 80 Zeichen/Zeile	PMODE TS		
Grafik-Betriebsart	PMODE GR	PLINE RLINE PMOVE RMOVE PHOME	HSET GPRINT AXIS PCIRCLE PCOLOR

Hinweis:
Die mit einem Stern (*) markierten Befehle können mit einem Dot-Matrix-Drucker (MZ-80P5 (K)) verwendet werden; andere Befehle sind nur mit einem Plotter verwendbar.

6.7 Maschinensprache-Steuerbefehle

PEEK

Format

PEEK <Adresse>

Erklärung

Diese Funktion gibt den Inhalt der spezifizierten Adresse als Dezimalzahl von 0 bis 255 an. <Adresse> kann eine Dezimalzahl von 0 bis 65535 oder eine vierstellige Dezimalzahl von $0000 bis $ FFFF sein.

Beispiel

Das folgende Programm zeigt die im Bereich von 40960 ($A000) bis 40975 ($A00F) im Speicher stehenden Daten an.

```
10 FOR AD=40960 TO 40975
20 PRINT PEEK (AD)
30 NEXT AD
40 END
```

POKE

Format

POKE <Adresse>,<Daten>[,<Daten>]...

Erklärung

Der POKE-Befehl schreibt eine Serie von Daten ab der spezifizierten Adresse. <Adresse> kann eine Dezimalzahl von 0 bis 65535 oder eine vierstellige Hexadezimalzahl $0000 bis $FFFF sein. <Daten> n kann im Bereich von 0 bis 255 oder $00 oder $FF liegen. Mit diesem Befehl kann man ungeachtet der durch den LIMIT-Befehl gesetzten Grenze Daten in jede beliebige Speicherstelle schreiben. Vorsicht ist geboten - durch nachlässige Verwendung dieses Befehls kann man den Monitor oder BASIC-Interpreter zerstören.

Beispiel

POKE $D000, $5F ..... Verwendet Hexadezimalzahlen.
POKE 53248,95 ..... Verwendet Dezimalzahlen. Die beiden obigen Befehle führen die gleiche Funktion aus.

INP@

Format	**INP@** <Port-Nummer>,<Variable>
Erklärung	Der INP@-Zeichenbefehl liest 8-Bit-Daten von dem mit <Port-Nummer> spezifizierten Eingangsport (=Eingangsanschluß), wandelt diese in eine Dezimalzahl um und weist sie der Variable zu. <Port-Nummer> kann im Bereich von 0 bis 127 liegen (hexadezimal $00 bis $7F). Die Port-Andressen 128 bis 255 ($80 bis $FF) sind für optionale Sharp-Peripherie-Geräte reserviert.

OUT@

Format	**OUT@** <Port-Nummer>,<numerischer Ausdruck>
Erklärung	Der OUT@-Zeichenbefehl dient zur Umwandlung der in <numerischer Ausdruck> spezifizierten Dezimalzahl (0 bis 255) in eine Zahl im Binarformat und überträgt diese an den durch <Port-Nummer> spezifizierten Ausgangsport (=Ausgangsanschluß). <Port-Nummer> kann im Bereich von 0 bis 12 liegen (hexadezimal $00 bis $7F). Die Port-Adressen 128 bis 255 (hexadezimal : $80 bis $FF) sind für optionale Sharp-Peripheriegeräte reserviert. Peripheriegeräte werden durch die an I/O-Ports übertragenen Daten gesteuert. Daher kann eine Betriebsstörung am Peripheriegerät auftreten, wenn man als <Port-Nummer> eine nicht zulässige Zahl spezifiziert.

USR

Format

USR (<Adresse>[, <Eingabe-Stringvariable>],[Ausgabe-Stringvariable>])

Abgekürztes Format

U.

Erklärung

Die USR-Funktion überträgt die Ablaufsteuerung an ein Maschinenspracheprogramm, das an der spezifizierten Adresse beginnt. Wie im Falle von CALL <Adresse> auch, erfolgt bei Auftreten von RET oder RET cc im Maschinenspracheunterprogramm Rücksprung zu der auf die USR-Funktion folgenden Anweisung. <Adresse> muß eine Dezimalzahl oder eine vierstellige Hexadezimalzahl sein. Wenn das Hauptprogramm die Ablaufsteuerung an das Maschinenspracheprogramm überträgt, werden die Parameter in die folgenden Register geladen:
DE-Register: Anfangsadresse von <Eingabe-Stringvariable> in den Speicher.
B-Register: Länge der <Eingabe-Stringvariable>
IX-Register: Adresse der Fehlerverarbeitungsroutine, falls angegeben.
Das Maschinenspracheprogramm legt bei Rückkehr zum Hauptprogramm die Verarbeitungsergebnisse in den folgenden Registern ab:
DE-Register: Anfangsadresse von <Ausgabe-Stringvariable> in den Speicher.
B-Register: Länge der <Ausgabe-Stringvariable>
Wenn Fehlerverarbeitung im Maschinenspracheprogramm benötigt wird, sind die folgenden Schritte erforderlich:

1. Angabe einer Fehlerverarbeitungsroutine im BASIC-Programm unter Verwendung einer ON ERROR GOTO-Anweisung.
2. Schreiben eines Programmsegments, das einen Fehlercode in das A-Register lädt und Sprung der Programmausführung zu der im IX-Register spezifizierten Adresse bewirkt.

LIMIT

Format

LIMIT <Adresse>
LIMIT MAX

Abgekürztes Format

LIM.

Erklärung

Der LIMIT-Befehl begrenzt den zur Verwendung durch BASIC verfügbaren Speicherplatz.
<Adresse> setzt die obere Grenze des BASIC-Bereiches; der Bereich , der auf diese Adresse bis $FEFF (65279) folgt, wird als Benutzerbereich reserviert. Der Bereich von $FF00 bis $FFFF wird vom Monitor als Arbeitsbereich verwendet, er kann daher nicht als Benutzerbereich verwendet werden. Die <Adresse> kann entweder eine Dezimalzahl oder eine vierstellige Hexadezimalzahl sein.
Um ein BASIC-Programm mit einem Maschinenspracheprogramm zu verbinden oder spezielle Daten in den Speicher einzulesen, muß genügend Speicherraum als Benutzerbereich reserviert werden.
Der LIMIT-Befehl muß am Anfang des Programms stehen. Der LIMIT MAX Befehl ist vorgesehen, um die durch LIMIT spezifizierte Grenze aufzuheben.

Beispiel

LIMIT $CFFF

Begrenzt den BASIC-Programmbereich auf $CFFF und legt den über diese Adresse hinausgehenden Bereich als Benutzerbereich fest.

LIMIT MAX

Zum Rücksetzen einer zuvor mit einem LIMIT-Befehl gesetzten Grenze.

6.8 Fehlerverarbeitungs-Befehle

ON ERROR GOTO

Format

ON ERROR GOTO { <Zeilennummer> / <Marke> }

Abgekürztes Format

ON ERR. G.

Erklärung

Die ON ERROR GOTO-Anweisung bewirkt bei Auftreten eines Fehlers Verzweigung zu der durch <Zeilennummer> oder <Marke> spezifizierten Zeile. Zur Steuerung der nachfolgenden Verarbeitung gemäß dem aufgetretenen Fehlertyp und gemäß der Nummer der Zeile, in der er auftrat, kann die ERN- oder ERL-Variable in einer Abfangroutine verwendet werden. Die ebenfalls enthaltene RESUME-Anweisung am Ende der Fehlerverarbeitungsroutine ermöglicht Rückkehr und Fortsetzung der Ausführung ab der Zeile, in der der Fehler auftrat. Durch Ausführung einer ON ERROR GOTO-Anweisung wird die durch die vorherige ON ERROR GOTO-Anweisung vorgegebene Fehlerverarbeitungs-Zeilennummer aufgehoben. Auch durch Ausführung eines CLR-Befehls wird die vorgegebene Fehlerverarbeitungs-Zeilennummer aufgehoben.

Beispiel

```
10  ON ERROR GOTO 100
20  INPUT "X=";X
30  PRINT SQR(X)
40  END
100 PRINT "FEHLER"
110 RESUME 20
```

Bei Auftreten eines Fehlers im obigen Programm erfolgt Anzeige der Meldung "FEHLER" und daraufhin die Rückkehr zur Zeile 20.

RESUME

Format

RESUME <Zeilennummer>
RESUME NEXT
RESUME 0
RESUME

Abgekürztes Format:

RESU.

Erklärung

Die RESUME-Anweisung dient zur Rückführung der Ablaufsteuerung von einer Fehlerverarbeitungsroutine an das Hauptprogramm. Das System bewahrt die Nummer der Zeile, in der der Fehler auftrat, im Speicher und bewirkt nach der Fehlerkorrektur die Rückführung der Programmausführung zu dieser oder einer sonstigen spezifizierten Zeile.
Die RESUME-Anweisung kann in einem der folgenden 4 Formate verwendet werden.

RESUME	Rückkehr zur Fehlerzeile
RESUME NEXT	Rückkehr zu der auf die Fehlerzeile folgenden Zeile.
RESUME <Zeilennummer>	Rückkehr zur Zeile <Zeilennummer>.
RESUME 0	Rückkehr zum Anfang des Hauptprogramms

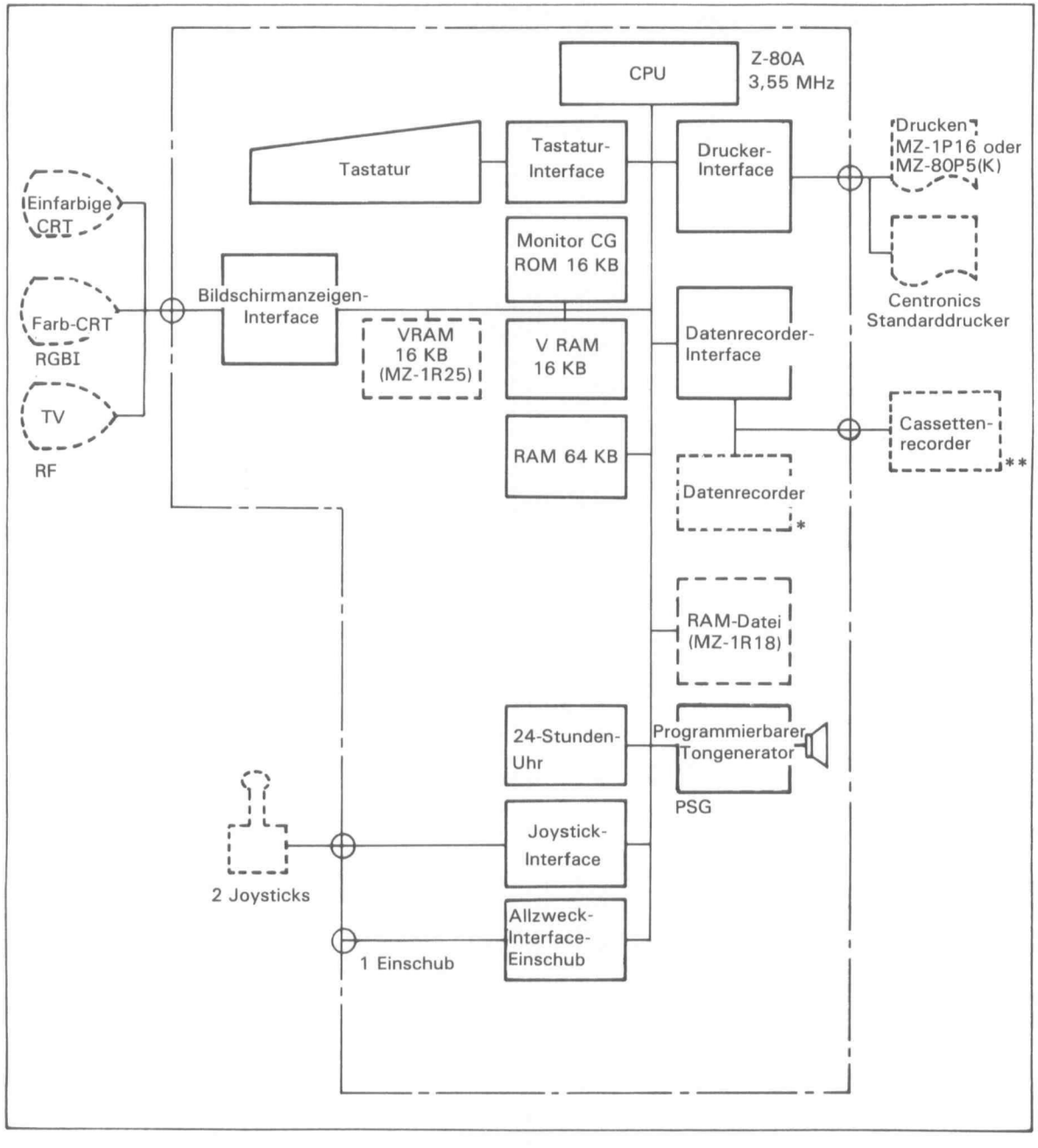
Zur Rückkehr von einer Fehlerverarbeitungsroutine zum Hauptprogramm auf jeden Fall die RESUME-Anweisung verwenden. Wenn RESUME zur Ausführung kommt, ohne daß ein Fehler aufgetreten ist, tritt ein Fehler auf.
Wenn RESUME nicht ausgeführt werden kann, tritt ein Fehler auf.

Kapitel 7 Hardware

In diesem Kapitel ist die Hardware des MZ-800 beschrieben. Desgleichen werden die an den MZ-800 anschließbaren Peripheriegeräte behandelt und die Verfahrensweise zu deren Anschluß.

7.1 Hardware des MZ-800

7.1.1 System-Konfiguration



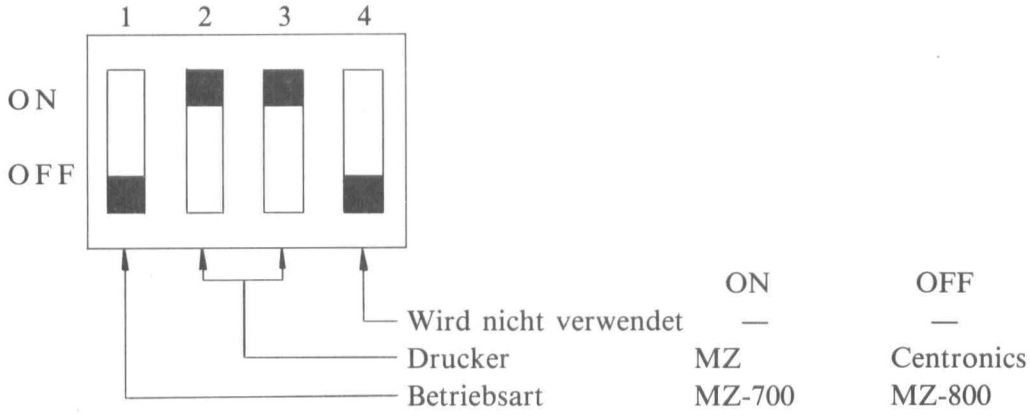
Optionales Gerät

* Standard mit dem MZ-821, optional mit dem MZ-811.
** Anschluß nur an den MZ-811 möglich.

7.1.2 Systemschalter-Einstellungen

Ein DIP-Schalter mit vier Positionen befindet sich an der Rückseite des Gerätes. Diesen Schalter nennen wir "Systemschalter". Die Funktionen seiner einzelnen Schalteinheiten sind wie folgt.

Hinweis:
Bei Einstellung des Systemschalters den Netzschalter ausschalten.



Schalter 1: Betriebsartenschalter
Dieser Schalter dient zur Umschaltung zwischen den Betriebsarten MZ-700 und MZ-800. Normalerweise läßt man den Schalter in der OFF-Stellung. (Für Hinweise zur MZ-700 Betriebsart, siehe Kapitel 9.)

ON: MZ-700 Betriebsart
OFF: MZ-800 Betriebsart

Schalter 2 und 3: Wahl der Drucker-Schnittstelle
Diese Schalter dienen zur Schnittstellen-Umschaltung zwischen dem MZ-Drucker-System und dem Centronics-System. Beide Schalter müssen in der gleichen Position stehen.

ON: MZ-Drucker
OFF: Centronics-Schnittstelle

Sollte Ihr Drucker nicht funktionieren, auch wenn beide Schalter auf OFF eingestelltsind, einen der beiden auf ON einstellen.

Schalter 4: Wird nicht verwendet

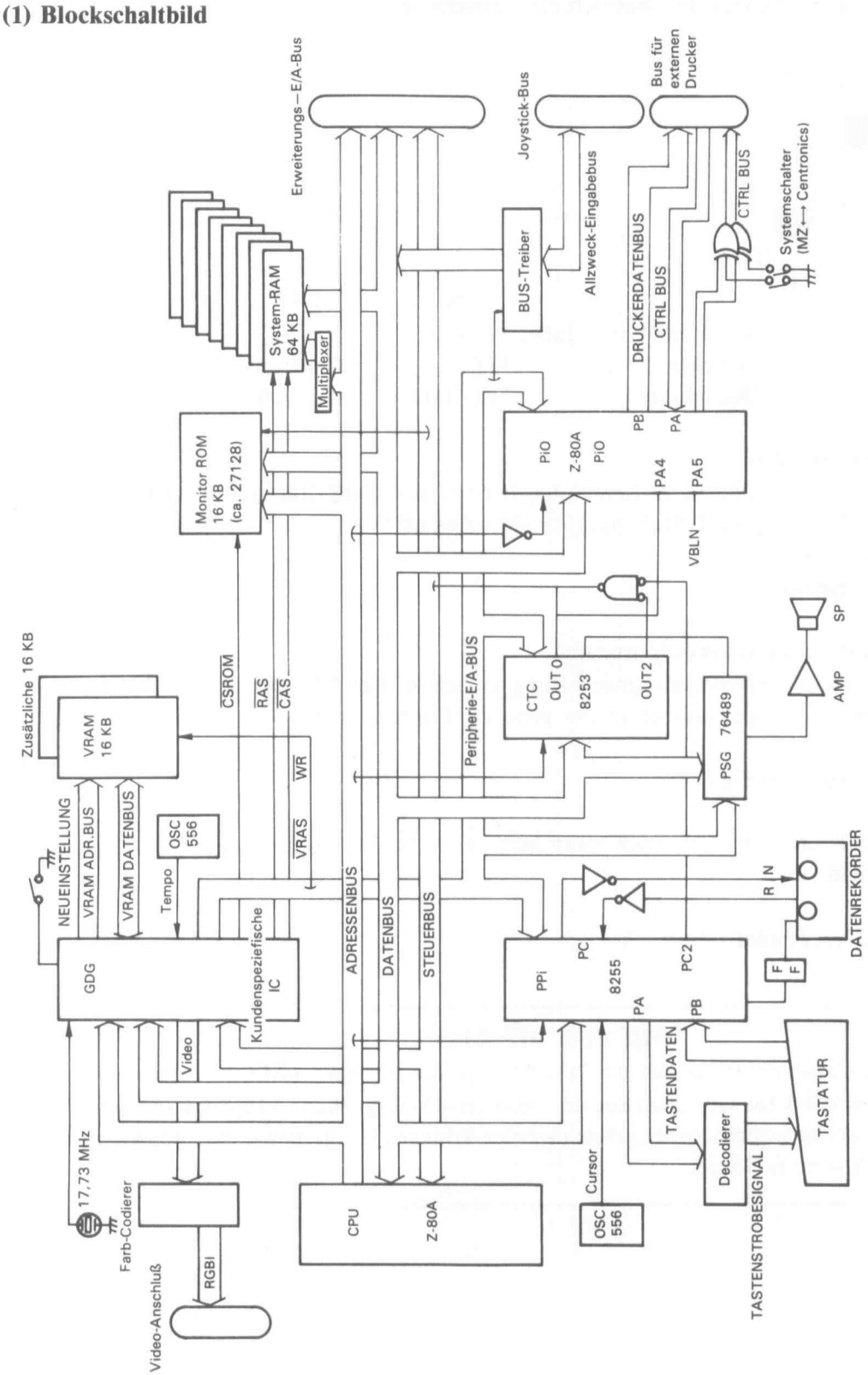
Mit dem MZ-811

Wenn ein normaler Cassettenrecorder an die Cassettenrecorder-Buchse des MZ-811 angeschlossen wird, dient dieser Schalter zur Umschaltung der Tonkopf-Polung. Wenn Programme oder Datendateien nicht vom Cassettenrecorder gelesen werden können, bringt man diesen Schalter versuchsweise in die andere Stellung.

7.1.3 I/O Port-Steuerung

Die folgenden Informationen dienen nur zur Bezugnahme. Bei Verwendung der I/O-Ports durch den Anwender auf jeden Fall mit gebührender Vorsicht verfahren, da möglicherweise Betriebsstörungen auftreten oder Peripheriegeräte betriebsunfähig werden, wenn der I/O-Port-Betrieb nicht korrekt ausgeführt wird.

(1) Blockschaltbild



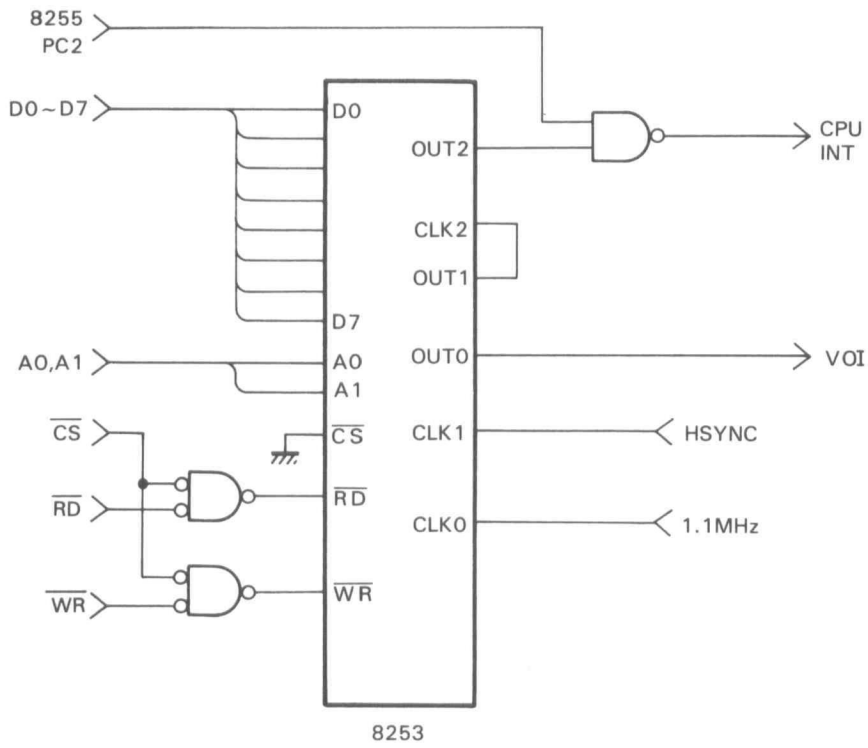
(2) I/O-Port-Adressen
Die folgenden I/O-Port-Adressen sind bereits den vorhandenen I/O-Geräten zugewiesen bzw. für geplante, anzuschließende Peripheriegeräte reserviert.

B0 bis B3 : Serieller I/O-Port
CC bis CF: GDG (Graphikanzeige-Generator)
D0 bis D3: 8255 (Cassettenrecorder -und Tastatursteuerung)
D4 bis D7: 8253 (programmierbarer Intervall-Taktgeber)
D8 bis DF: FDC (Floppydisk-Controller)
E0 bis E6 : GDG
F0 bis Fl : Joystick-Eingänge
F2 : PSG-Ausgang
F4 bis F7 : QDC
FC bis FF: Z-80A PIO (Drucker-Steuerung)

(3) Programmierbarer Taktgeber (8253)
Der MZ-800 hat einen eingebauten programmierbaren Intervall-Taktgeber. Dieser Taktgeber dient zur Steuerung der eingebauten Uhr und des programmierbaren Tongenerators. In der MZ-700-Betriebsart werden die "memory mapped" I/O-Adressen $E004 bis $E007 diesem Taktgeber zugewiesen; in der MZ-800-Betriebsart werden dem Taktgeber die Adressen für isolierte Ein/Ausgabe $D4 bis $D7 zugewiesen.

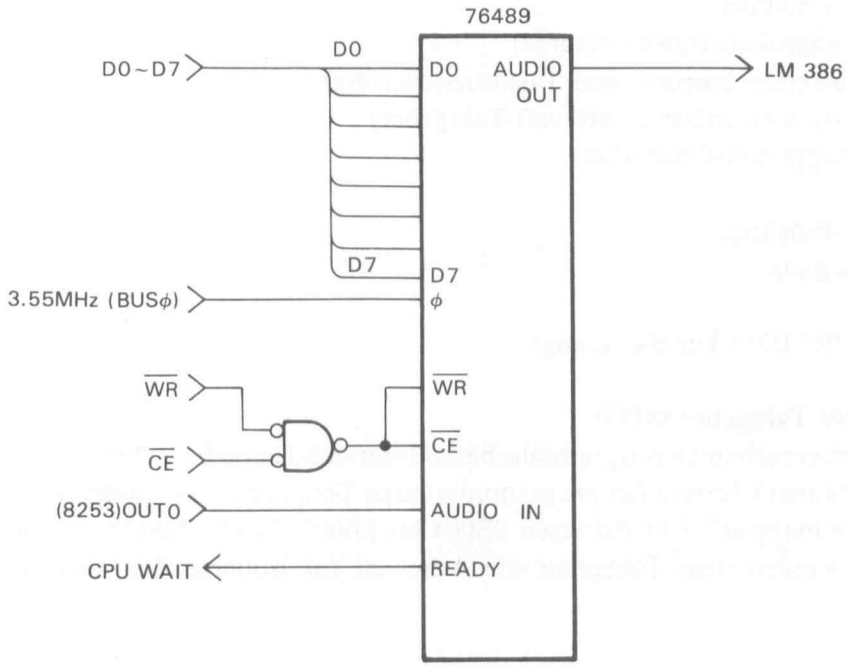
$D4 : Zähler 0
$D5 : Zähler 1
$D6 : Zähler 2
$D7 : Steuerwortregister

Zähler 0 wird für den programmierbaren Tongenerator verwendet, Zähler 1 intern und Zähler 2 zur Unterbrechung der CPU.



(4) Programmierbarer Tongenerator (76489)

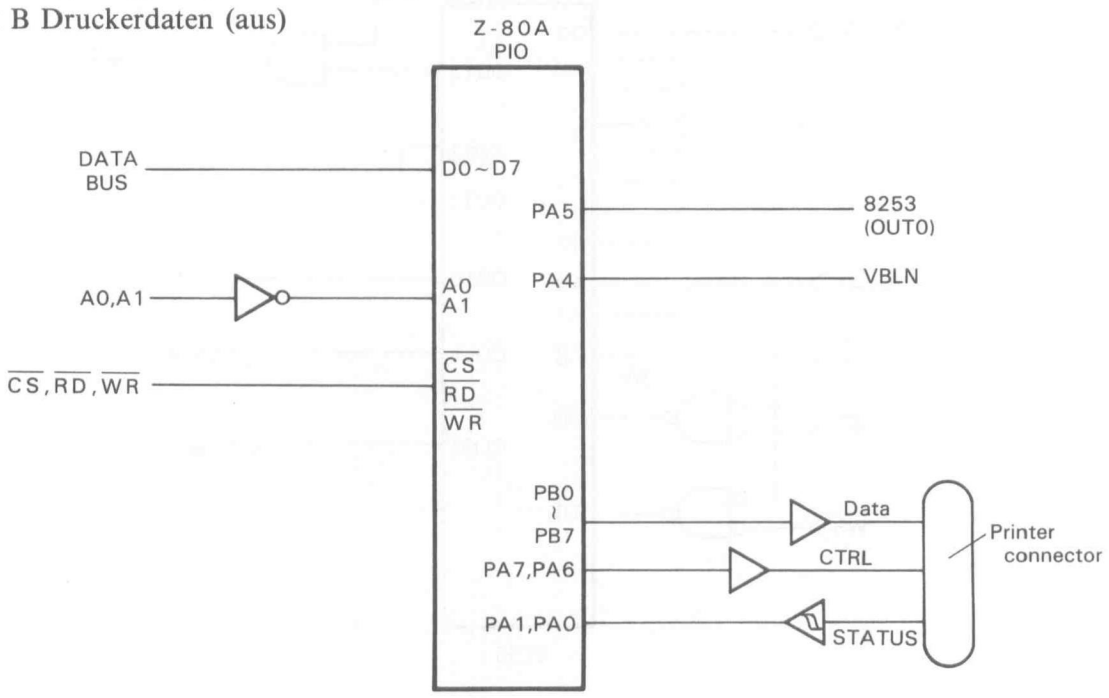
Der MZ-800 hat einen eingebauten programmierbaren Tongenerator (PSG) zur Erzeugung von Dreiklang-Akkorden über 8 Oktaven. Die I/O-Portadresse $F2 wird dem PSG zugewiesen. Für weitere Einzelheiten zur Steuerung siehe Anhang .



(5) Drucker-Schnittstelle (Z-80A PIO)

Als Drucker-Schnittstelle verwendet der MZ-800 eine Z-80A PIO. Die I/O-Portadressen von $FC bis $FF werden der Z-80A PIO zugewiesen.

$FC: Steuerregister
$FD: Steuerregister
$FE: Port A
PA0, PA1: Druckerzustand (ein)
PA4, PA5: Systemzustand (ein)
PA6, PA7: Druckersteuerung (aus)
$FF: Port B Druckerdaten (aus)



(6) Tastatur -und Datenrecorder-Controller (8255)

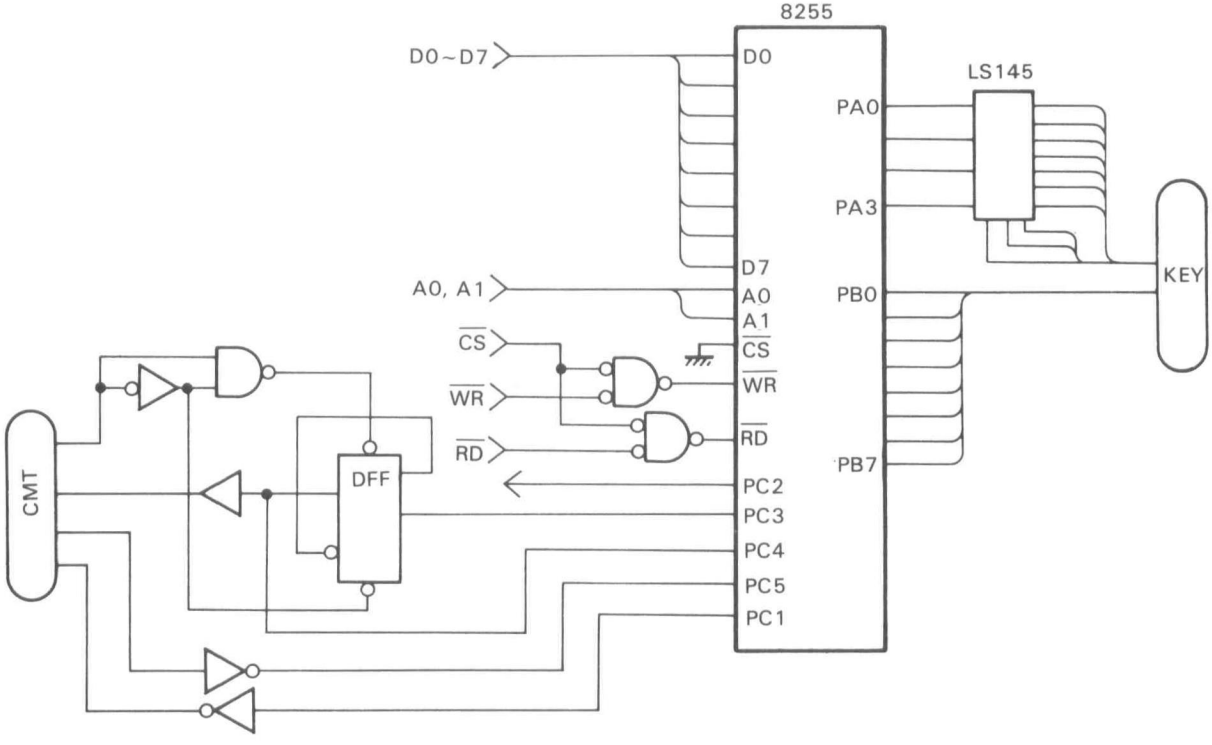
Zur Steuerung von Tastatur und Datenrecorder verwendet der MZ-800 eine 8255.
In der MZ-700-Betriebsart werden diesem Taktgeber die memory mapped I/O-Adressen $E000 bis $E003 zugewiesen; in der MZ-800-Betriebsart werden ihm die Adressen für isolierte Ein/Ausgabe $D0 bis $D3 zugewiesen.

$D0: Port A
- PA0 bis PA3: KEYSTROBE-Signale (aus)
- PA4 bis PA5: JOYSTROBE-Signale (aus)
- PA7: Cursor RST (aus)

$D1: Port B
- PB0 bis PB7: KEYDATA-Signale (ein)

$D2: Port C
- PC0: SOUND MASK (aus)
- PC1: CMTWR (aus)
- PC2: INT (aus)
- PC3: MOTOR (aus)
- PC4: SENSE (ein)
- PC5: CMTRD (ein)
- PC6: Cursor FLSH (ein)
- PC7: VBLNK (ein)

$D3: Steuerregister



7.2 Peripheriegeräte

Einige der hier beschriebenen peripheriegeräte sind möglicherweise nicht in Ihren Land erhältlich Bei Anschluß von Peripheriegeräten unbedingt die Stromversorgung abschalten.

7.2.1 Standard-Schnittstellen

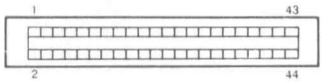
Der MZ-800 ist mit den folgenden Schnittstellen ausgestattet.

- Bildschirmanzeige-Schnittstelle
- Tastatur-Schnittstelle
- Cassettenrecorder-Schnittstelle
- Drucker-Schnittstelle
- Joystick-Schnittstelle

7.2.2 I/O-Erweiterungs-Anschlußbuchse

Eine I/O-Erweiterungs-Anschlußbuchse ist innerhalb des Computers vorgesehen; durch Entfernen der Erweiterungsschlitz-Abdeckung von der Rückwand wird diese Buchse zugänglich. Diese Buchse ist für den Anschluß einer optionalen Schnittstelle oder Erweiterungseinheit vorgesehen. Die Stiftbelegungen der Erweiterungs-I/O-Anschlußbuchse sind wie folgt.

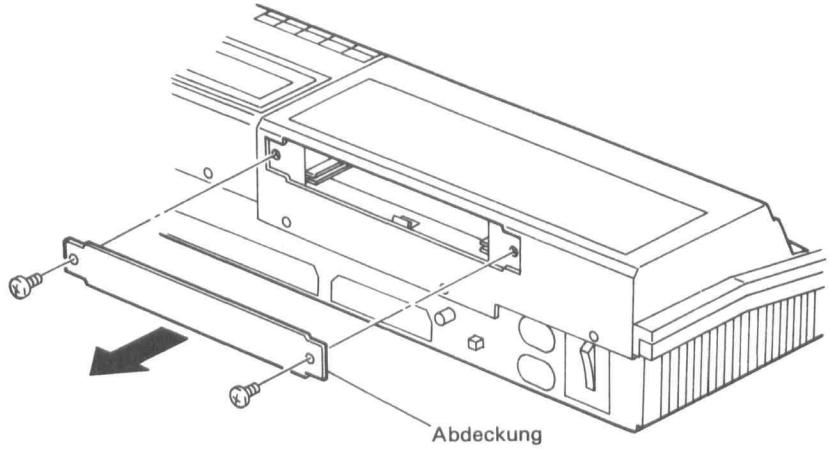
A	Bestückungsseite		Lötseite
1	+5V	2	+5V
3	D2	4	D3
5	D1	6	D4
7	D0	8	D5
9	GND	10	D6
11	ADF	12	D7
13	ADE	14	BUS Ø
15	ADD	16	$\overline{M1}$
17	ADC	18	$\overline{WR}$
19	ADB	20	$\overline{RD}$
21	ADA	22	$\overline{IORQ}$
23	AD9	24	$\overline{MREQ}$
25	AD8	26	GND
27	AD7	28	$\overline{HALT}$
29	AD6	30	IEI
31	AD5	32	NC.
33	AD4	34	RESET
35	AD3	36	$\overline{EXRESET}$
37	AD2	38	$\overline{INT}$
39	AD1	40	$\overline{EXWAIT}$
41	AD0	42	NC.
43	GND	44	GND



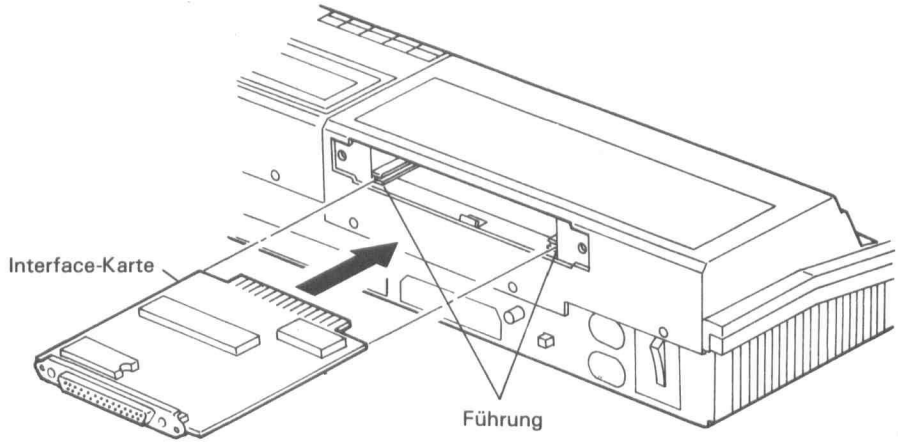
Anschluß

• Einbau einer optionalen Schnittstelle

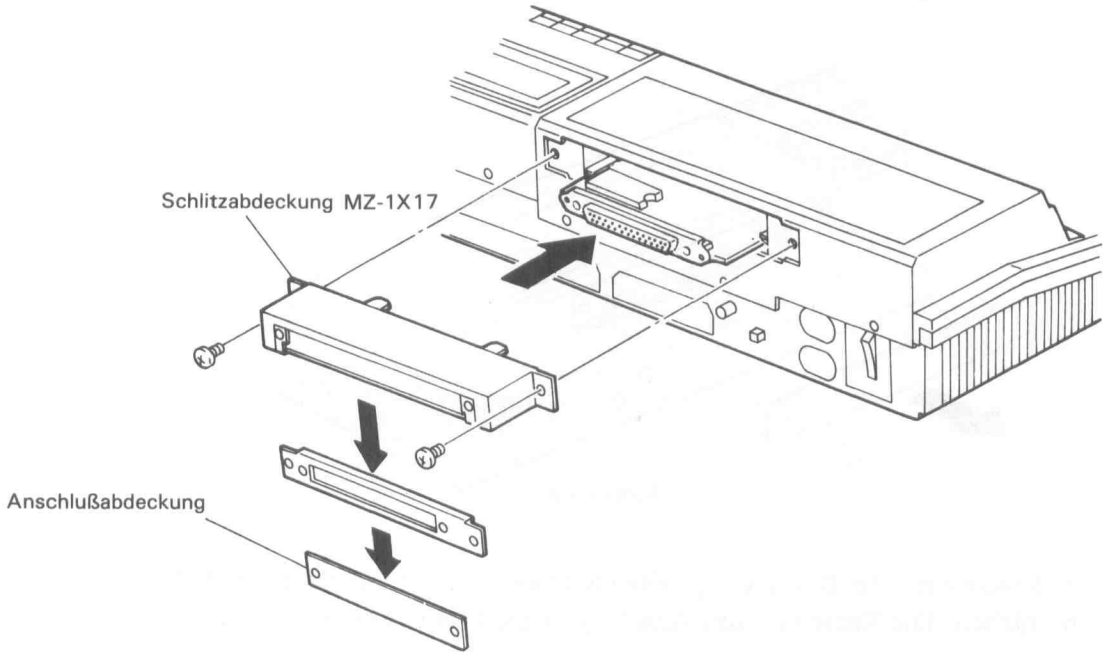
1) Die Erweiterungsschlitz-Abdeckung entfernen. (Die Abdeckung aufbewahren, für den Fall, daß die Schnittstelle wieder ausgebaut wird.)



2) Die Schnittstellenkarte mit der Bestückungsseite nach oben gerichtet entlang den Führungsschienen ins Gerät einschieben. Die Karte bis zum Anschlag in die I/O-Erweiterungs-Anschlußbuchse einschieben.



3) Die Anschlußabdeckung von der optionalen Schlitzabdeckung MZ-1X17 entfernen. Die optionale Schlitzabdeckung an der Rückwand des MZ-800 anbringen.

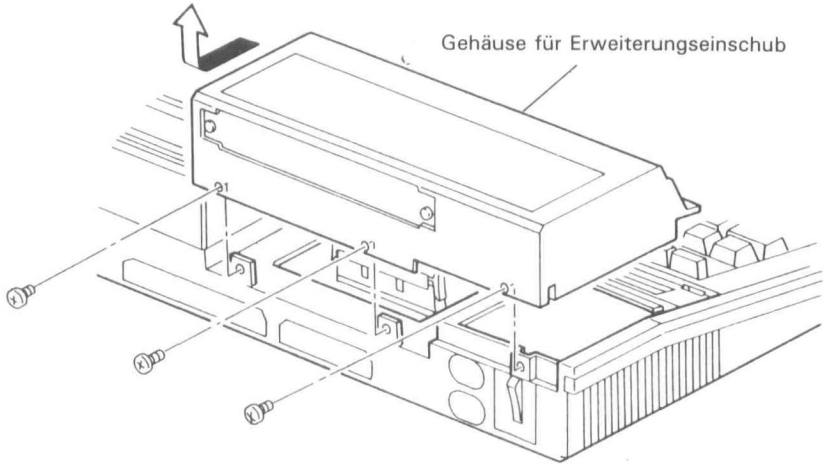


Schlitzabdeckung MZ-1X17 Anschlußbuchsenabdeckung Rückwand für weitere Hinweise siehe Handbuch für die eingebaute Schnittstelle.

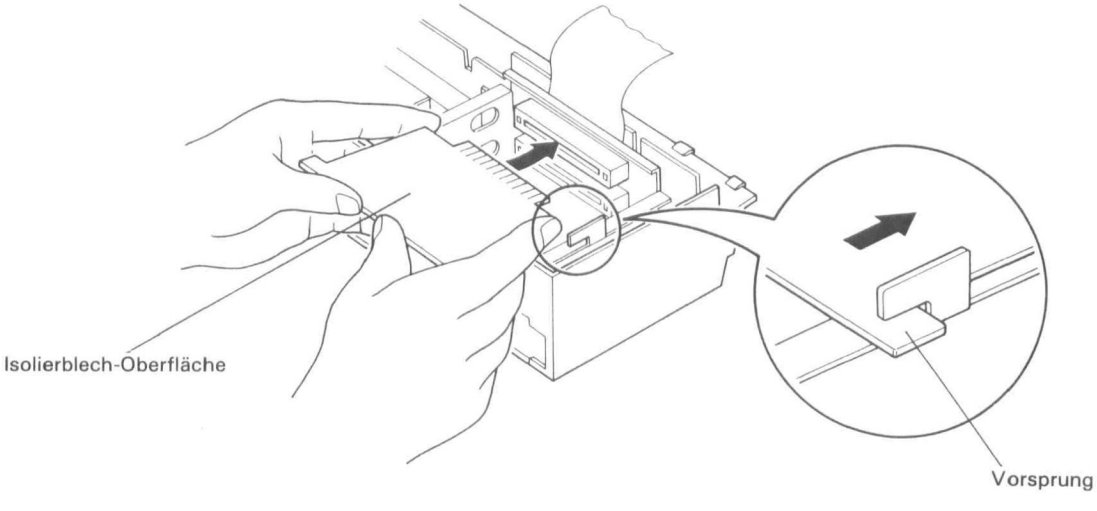
7.2.3 RAM-Datei-Steckkarte (MZ-1R18)

Die RAM-Datei-Steckkarte ist ein Speichergerät, das grundsätzlich in der gleichen Weise verwendet werden kann wie ein Diskettenlaufwerk, abgesehen von der Tatsache daß beim Ausschalten der Stromversorgung der gesamte Speicherinhalt verlorengeht. Für weitere Einzelheiten zur RAM-Datei-Steckkarte siehe zugehöriges Handbuch.
Zum Einbau der RAM-Datei-Steckkarte bitte folgende Hinweise beachten.

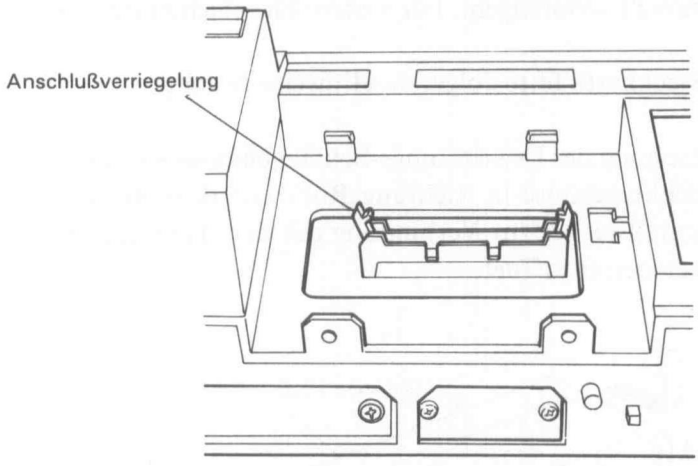
(1) Die drei Schrauben zur Befestigung des Erweiterungs-Schlitzgehäuses wie nachstehend gezeigt entfernen. Das Erweiterungs-Schlitzgehäuse in Richtung Rückseite drücken und vom rückwärtigen Teil her abheben. Den Anschlußstecker zur Verbindung mit dem Hauptgerät ziehen, indem man zunächst die Anschlußsteckersperren öffnet.



(2) Das Erweiterungs-Schlitzgehäuse mit der Oberseite nach unten gerichtet auf eine flache Unterlage legen, so daß die zwei darin befindlichen Anschlußbuchsen sichtbar werden. Nun die RAM-Datei-Steckkarte mit der Lötseite nach oben in die obere Anschlußbuchse einstecken. Auf jeden Fall sicherstellen, daß der Vorsprung der Steckkarte in den Haken des Erweiterungs-Schlitzgehäuses einrastet (siehe nachstehende Zeichnung).



(3) Den im Schritt 1 von der Buchse des Hauptgerätes abgetrennten Stecker wieder einstecken. Das Erweiterungs-Schlitzgehäuse wieder anbringen und mit den drei Schrauben wieder befestigen. Um den Stecker sicher zu befestigen, die Steckersperren auf jeden Fall wieder schließen.



Hinweis:
Wenn das Kabel zwischen Gehäuse und Hauptgerät eingeklemmt wird, läßt sich das Erweiterungs-Schlitzgehäuse nicht einwandfrei anbringen.

7.2.4 Joystick

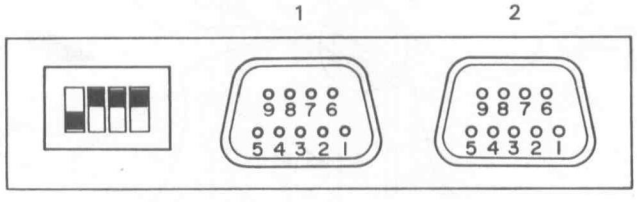
Von Atari, Inc. oder anderen. Herstellern gebaute Joysticks können mit dem MZ-800 verwendet werden.

JOYSTICK 1

1	FWDA
2	BACKA
3	LEFTA
4	RIGHTA
5	+5V
6	TRG1A
7	TRG2A
8	COMA
9	GND

JOYSTIC 2

1	FWDB
2	BACKB
3	LEFTB
4	RIGHTB
5	+5V
6	TRG1B
7	TRG2B
8	COMB
9	GND

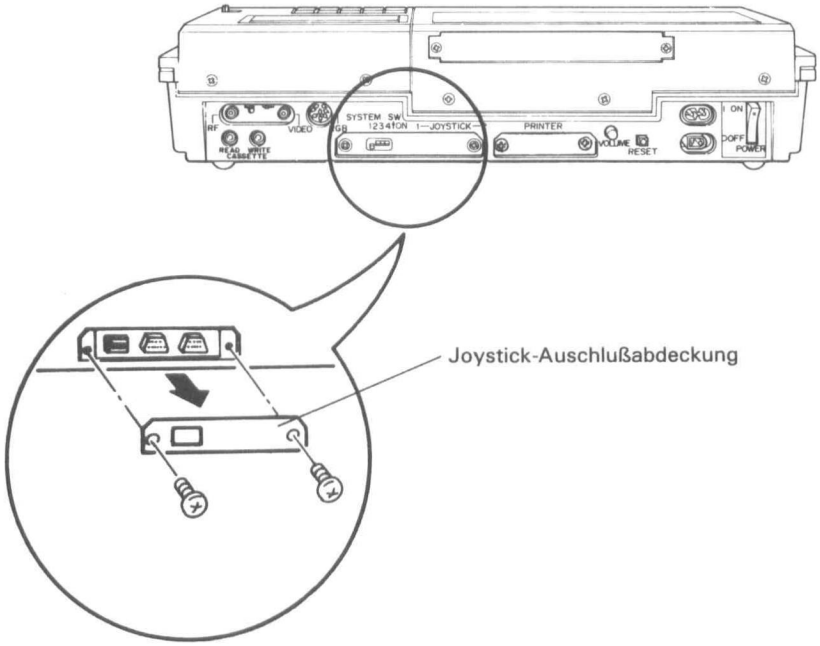


Anschluß

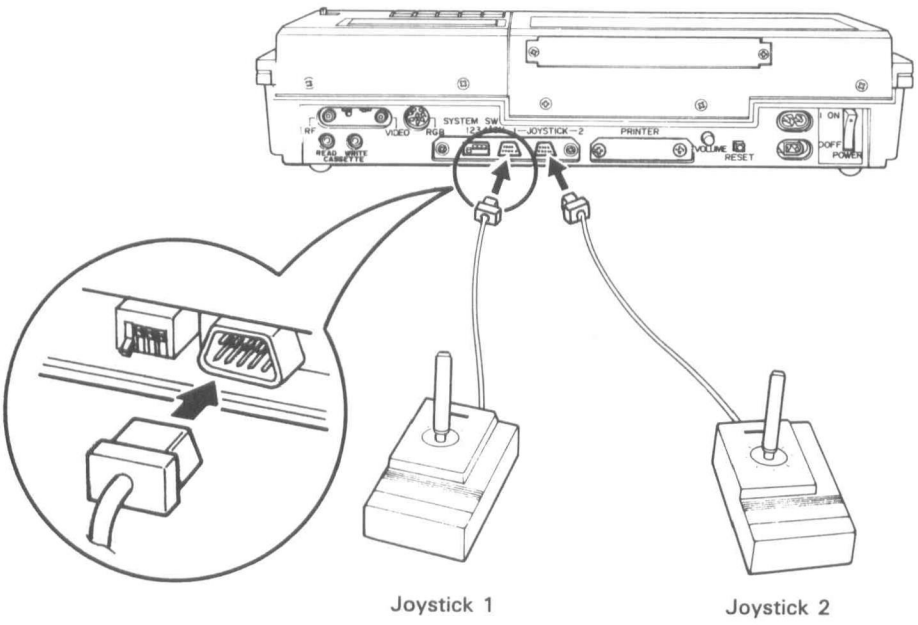
Hinweis:
Analoge Joysticks können nicht verwendet werden!

Das Anschlußverfahren ist wie folgt.

1) Die Joystick-Anschlußbuchsen-Abdeckung von der Rückseite entfernen. Die Abdeckung aufbewahren, für den Fall, daß die Joysticks abgetrennt werden sollen.



2) Die Kabel der Joysticks wie nachstehend gezeigt anschließen.



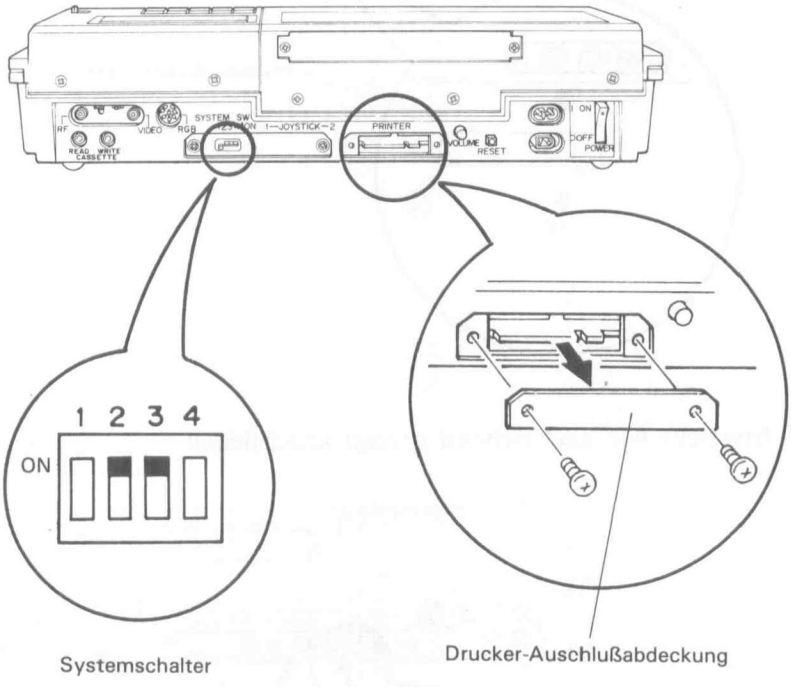
7.2.5 Drucker

Mit dem MZ-800 können verschiedene Ausführungen von Druckern verwendet werden. SHARP bietet zwei verschiedene Ausführungen für den MZ-800 an.

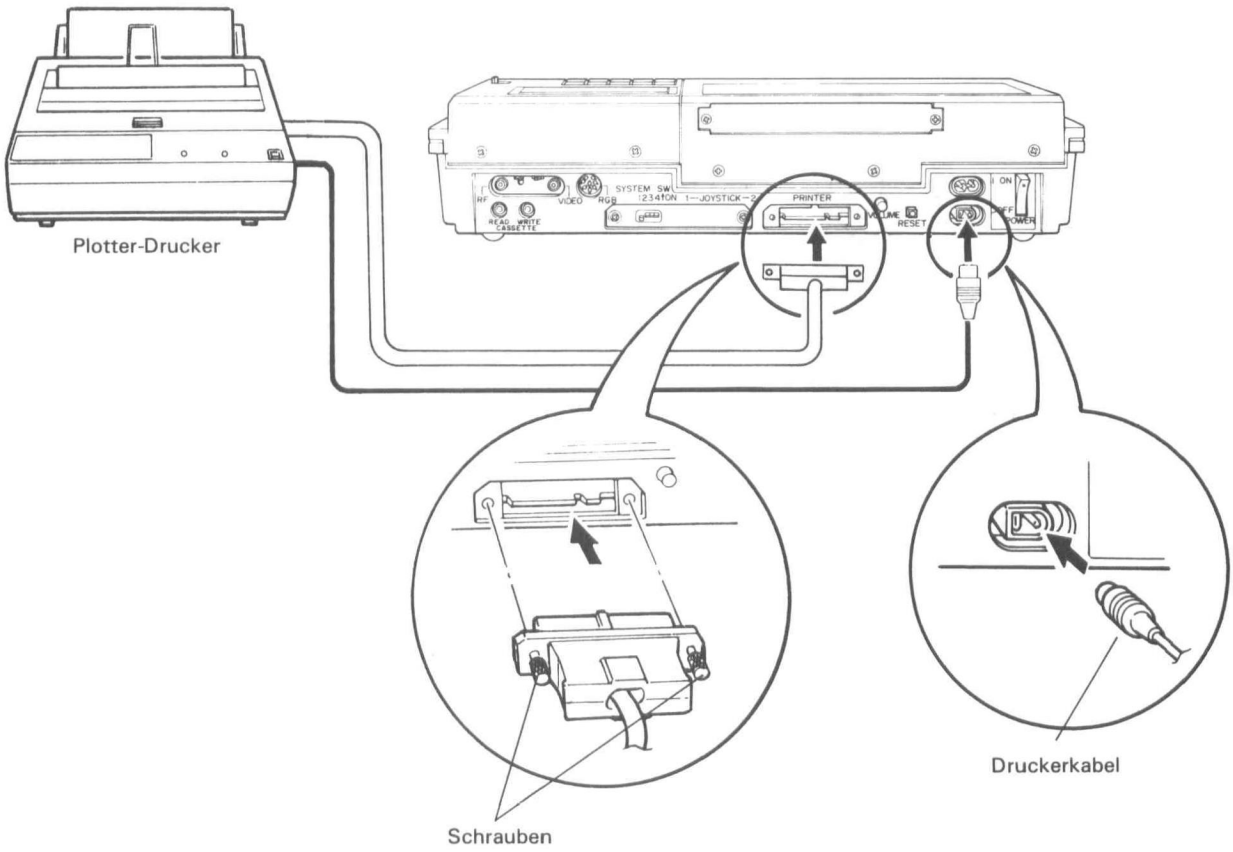
(1) Plotter MZ-1P16

Das Anschlußverfahren ist wie folgt.

1) Die Schalter 2 und 3 des Systemschalters auf ON stellen.
2) Die beiden Schrauben zur Befestigung der Drucker-Anschlußbuchsen-Abdeckung an der Rückseite des MZ-800 entfernen. (Die Abdeckung aufbewahren, für den Fall, daß das Druckerkabel wieder abgetrennt werden soll.



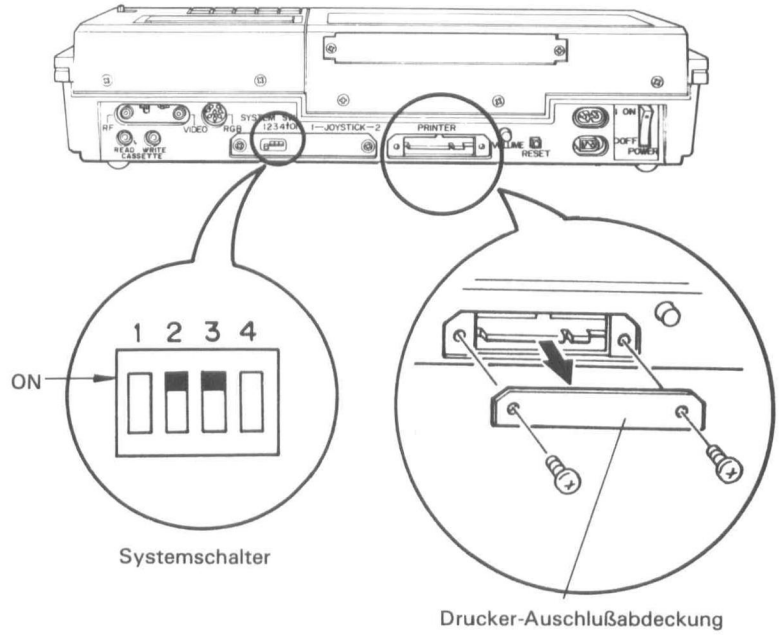
3) Den Stecker des Druckerkabels an die Kartenrandkontakte des MZ-800 anschließen, wobei die Taste des Steckers nach oben gerichtet ist und den Stecker mit Schrauben am MZ-800 befestigen.



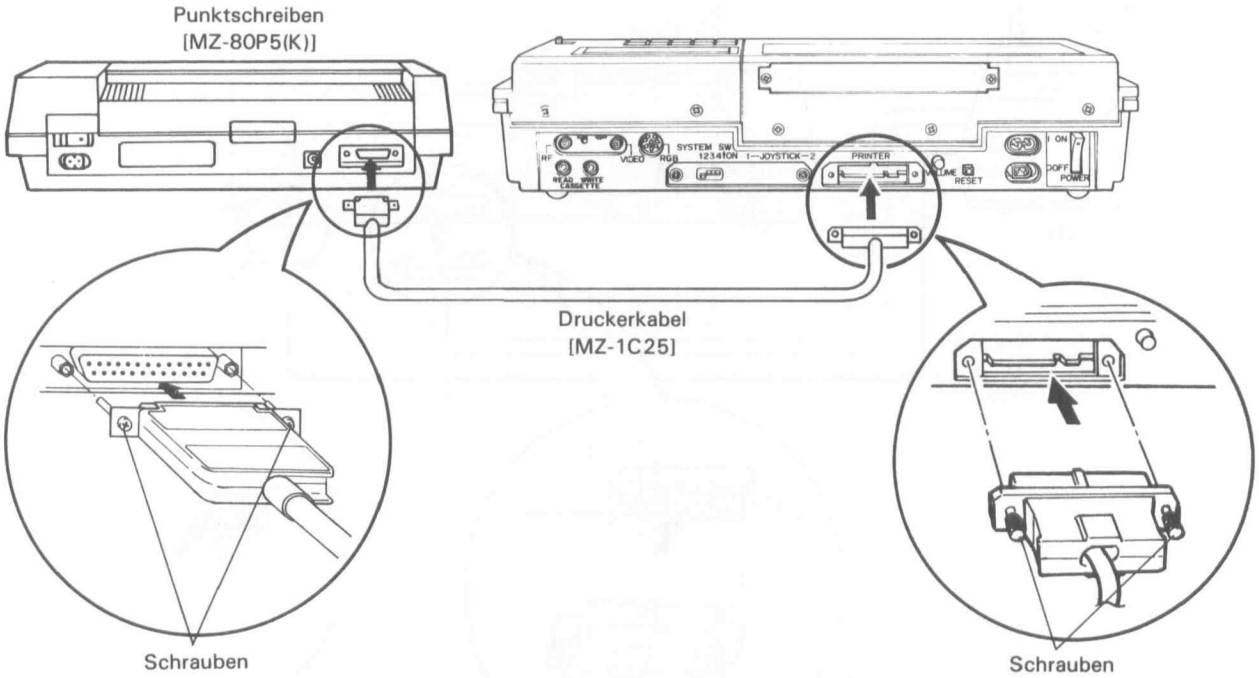
4) Das Druckernetzkabel an die Plotter-Netzbuchse an der Rückseite des MZ-800 anschließen.

(2) Dotmatrixdrucker MZ-80P5(K)

1) Die Schalter 2 und 3 des Systemschalters auf ON stellen.
2) Zur Entfernung der Drucker-Steckerabdeckung von der Rückseite des MZ-800 die beiden Schrauben entfernen. (Die Abdeckung aufbewahren, falls das Druckerkabel wieder abgetrennt werden soll.

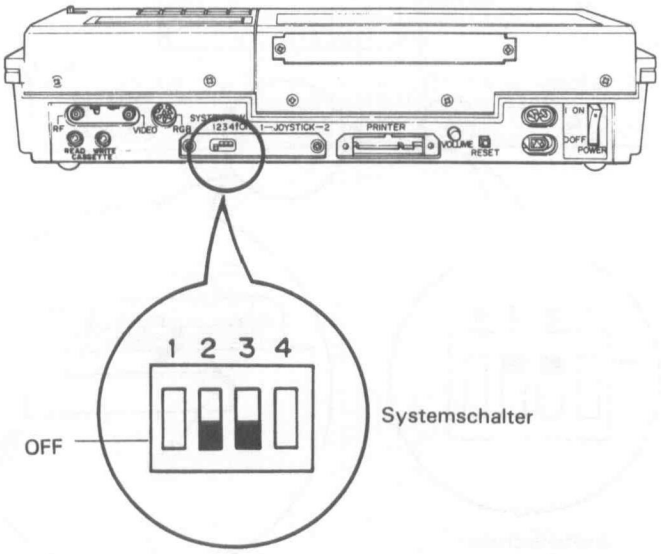


3) Den Anschlußstecker des Druckers mit dem optionalen Verbindungskabel MZ-lC25 am Druckerstecker des MZ-800 anschließen. Den Stecker auf jeden Fall mit Schrauben am MZ-800 befestigen.



(3) Sonstige Drucker

Durch Einstellung der Schalter 2 und 3 des Systemschalters auf OFF wird der Anschluß eines mit Centronics-Schnittstelle ausgestatteten Druckers ermöglicht. Einige vom Hersteller als Drucker mit Centronics-Normschnittstelle ausgestattete Geräte gehorchen jedoch effektiv nicht dem Centronics-Standard; solche Drucker können nicht verwendet werden. Bei einigen Druckern unterscheiden sich die Codesätze von denen für den Drucker MZ-1P16 oder MZ-80P5(K). Solche Drucker können verwendet werden, es ist jedoch möglicherweise eine spezielle Programmierung erforderlich, um sie optimal nutzen zu können.



Drucker-Interface

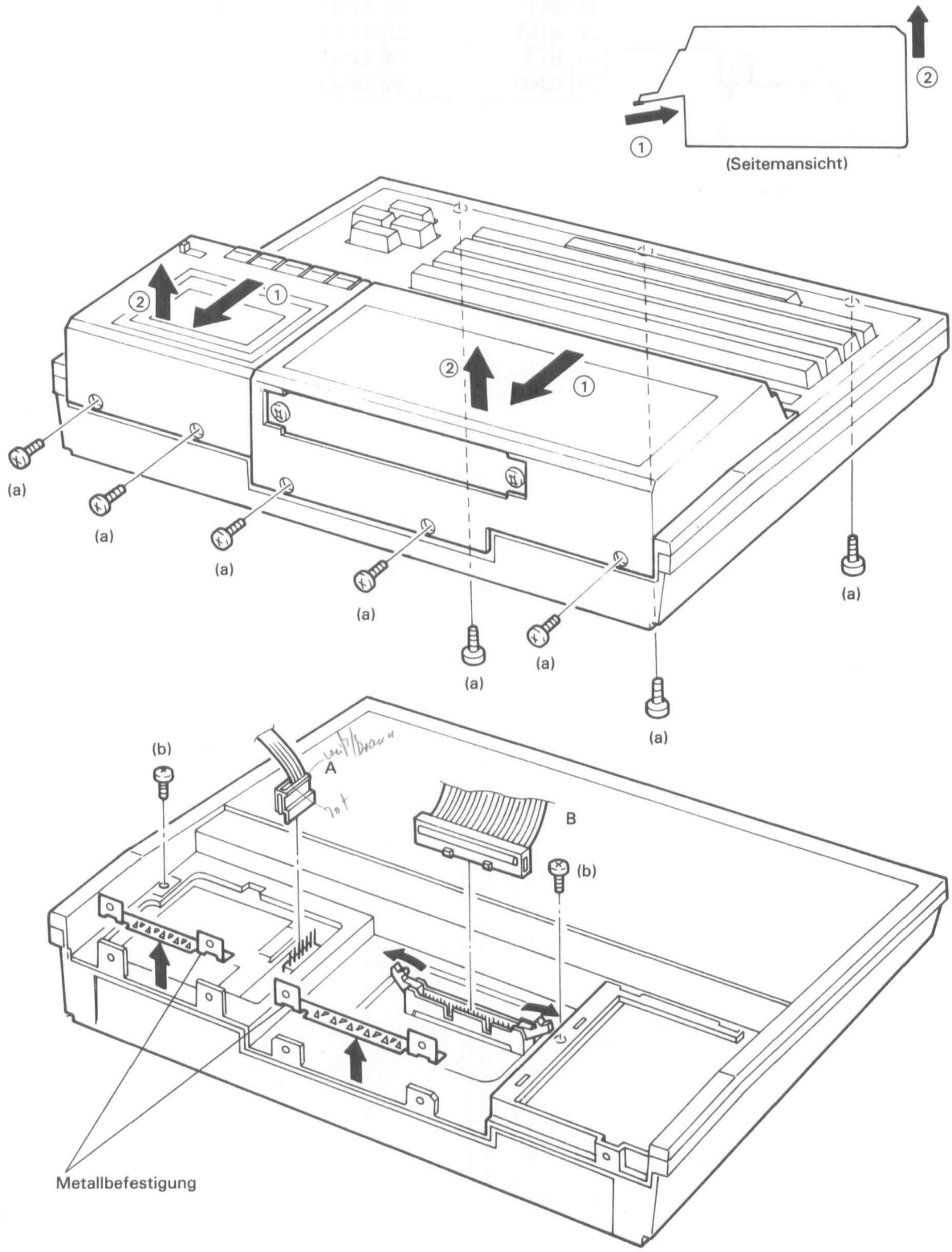
Component side		Solder side	
1	RDP	2	GND
3	RD1	4	GND
5	RD2	6	GND
7	RD3	8	GND
9	RD4	10	GND
11	RD5	12	GND
13	RD6	14	GND
15	RD7	16	GND
17	RD8	18	GND
19	IRT	20	GND
21	$\overline{\text{RDA}}$	22	GND
23	$\overline{\text{STA}}$	24	GND
25	GND	26	GND

Anschluß

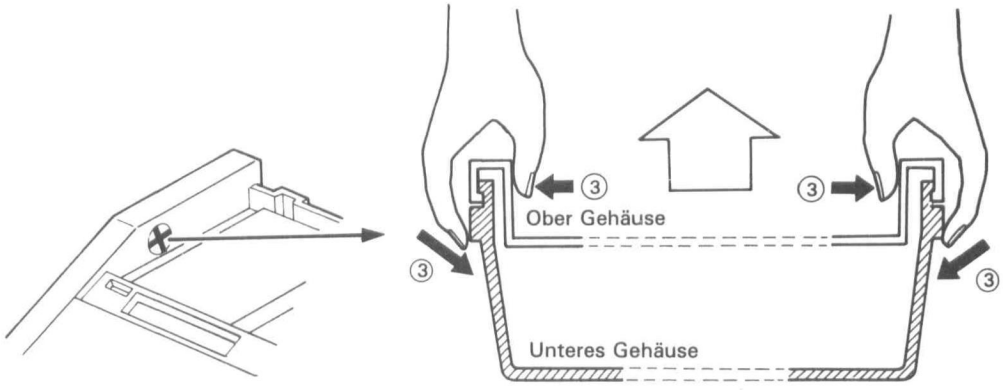
7.2.6 Optionaler Grafikspeicher MZ-1R25

Der optionale Grafikspeicher MZ-1R25 verbessert weiterhin die Anzeigefähigkeit des MZ-800. Dazu gehören zwei IC-Chips, die wie folgt im Gehäuse eingebaut werden.

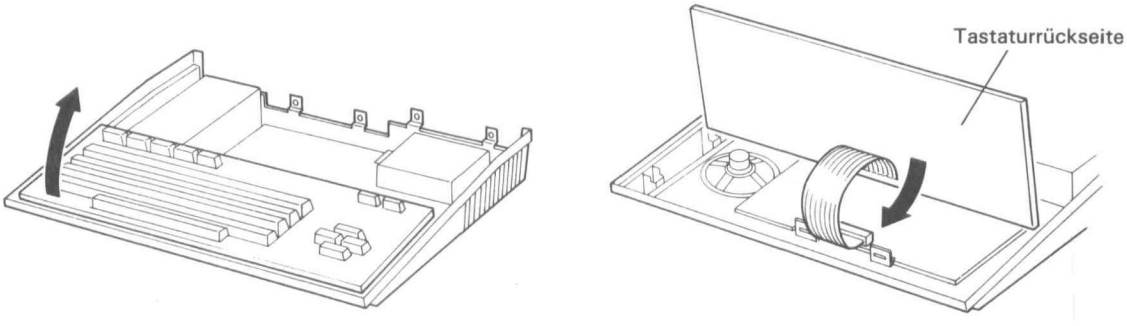
1) Die Schrauben (a) entfernen, dann Cassettenrecorder und Erweiterungsschlitz-Fachdeckel wie durch die Zeilen ① und ② gezeigt abnehmen.
2) Die Anschlußstecker A und B abziehen.
3) Die Schrauben (b) entfernen.
4)Die Metallbefestigung abnehmen.



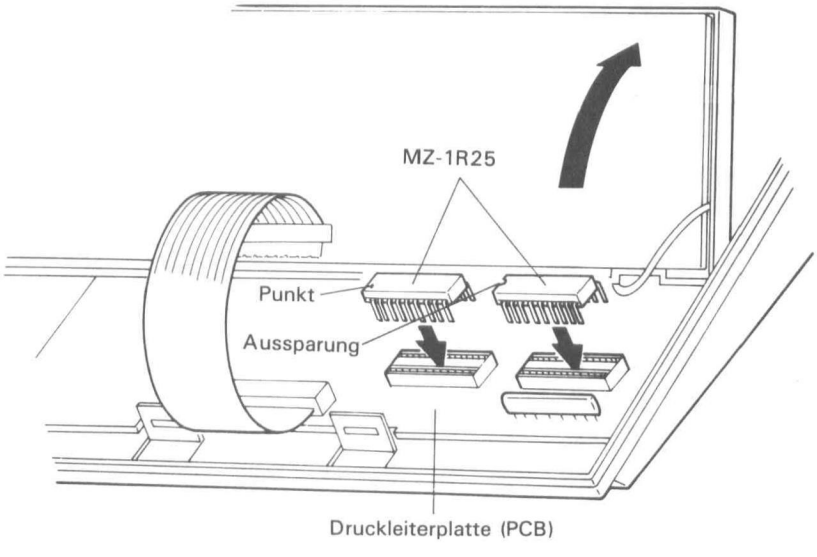
5) Das obere Gehäuse an den mit den Pfeilen ③ gezeigten Teilen eindrücken und halten, dann zum Abtrennen des oberen Gehäuses vom unteren Gehäuse das obere Gehäuse abziehen.



6) Die Vorderseite der Tastatur anheben und wie nachstehend gezeigt aufklappen.



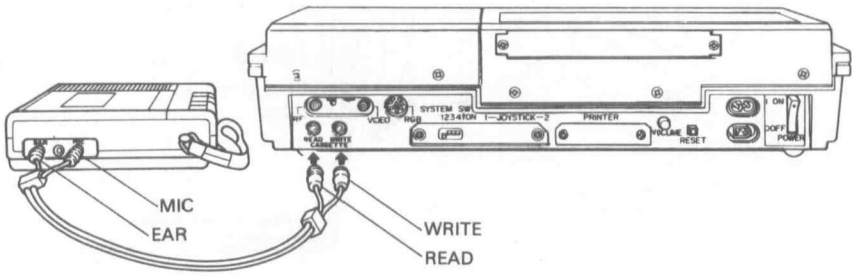
7) Zwei IC-Halterungen befinden sich in der rechten vorderen Ecke der Hauptleiterplatte. Die IC-Chips wie nachstehend gezeigt in die beiden IC-Halterungen einsetzen. Dabei auf die Einbaurichtung achten, da durch Einsetzen in umgekehrter Richtung ihre internen Schaltkreise zerstört werden.



8) Um die Gehäuse wieder anzubringen, die obigen Schritte nochmals in umgekehrter Reihenfolge ausführen.

7.2.7 Externer Cassettenrecorder (nur für MZ-811)

Mit dem MZ-811 kann man einen gewöhnlichen Cassettenrecorder als Datenrecorder verwenden. Schließen Sie die WRITE und READ Buchsen an der Rückseite des MZ-811 an die MIC und EAR (bzw. EXT.SP) Buchsen des Cassettenrecorders an. Zum Anschluß ein abgeschirmtes Audio-Verbindugskabel mit 3,5 ϕ Buchsen verwenden.



Bei Verwendung eines gewöhnlichen Cassttenrecorders bitte die folgenden Punkte beachten:

1) Die Meldung " ⬇ RECORD.PLAY" erscheint nicht, wenn ein SAVE Befehl eingegeben wird. Vor der Eingabe dieses Befehls auf jeden Fall die Aufnahmetaste [RECORD] drücken. Zum Stoppen des Cassettenrecorders nachdem die Meldung "Ready" angezeigt wird, die [STOP] -Taste drücken. Nur durch Drücken der [STOP] -Taste kann der Cassettenrecorder gestoppt werden.

2) Die Meldung " ⬇ PLAY" erscheint nicht, wenn ein LOAD Befehl eingegeben wird. Vor der Eingabe des Befehls auf jeden Fall die Wiedergabe-Funktion aktivieren.
Zum Stoppen des Cassettenrecorders nachdem die Meldung "Ready" angezeigt wird, die [STOP] -Taste drücken. Nur durch Drücken der [STOP] -Taste kann der Cassettenrecorder gestoppt werden.

3) Die Lautstärke- und Klangsteller des Cassettenrecorders müssen korrekt eingestellt werden. Bestimmte Modelle (z.B. solche mit Aussteuerungsautomatik) können möglicherweise nicht verwendet werden. In diesem Falle empfehlen wir den Kauf des Datenrecorders MZ-1T04.

4) Das Laden von Programmen ist nur möglich, wenn die Tonkopf-Polung korrekt ist. Wenn sich das Programm nicht laden läßt, bringen Sie versuchsweise den Schalter 4 des System-Schalters zur Umschaltung der Tonkopf-Polung in die andere Stellung.

5) Für das Einlesen von Programmen (Laden vom Band) auf jeden Fall den gleichen Cassettenrecorder verwenden wie für die Aufnahme. Bei Verwendung eines anderen Cassettenrecorders ist ein einwandfreier Betrieb gegebenenfalls nicht möglich.

Kapitel 8 Monitor

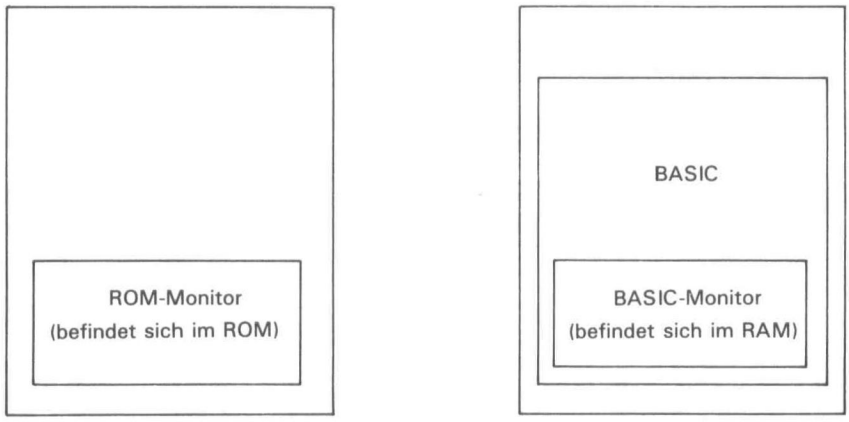
8.1 Allgemeines

Obwohl ein Maschinensprache-Programm schwer zu verstehen ist, hat es viele Vorteile, z.B. ist die Ausführung wesentlich schneller und es wird erheblich weniger Speicherplatz verbraucht als bei einem BASIC-Programm. Darüberhinaus kann man mit Maschinenspräche Programme schreiben, die die Eigenschaften der Hardware in vollem Umfang und optimal nutzen. Zur Erstellung von Maschinensprache-Programmen verwenden wir Monitorbefehle. In diesem Kapitel werden die Funktionen und Verwendungsweisen der einzelnen Monitorbefehle beschrieben.
Bei Verwendung eines Monitorbefehls bitte die folgenden Punkte beachten.

* Die Eingabe jedes Monitorbefehls wird durch Drücken der [CR]-Taste abgeschlossen.
* Alle Befehle müssen genau in dem in diesem Handbuch beschriebenen Format eingegeben werden. In der Befehlszeile keine Leerstellen eingeben.
* Ein Byte lange Daten in einem Monitorbefehl müssen mit einer zweistelligen Hexadezimalzahl spezifiziert werden, 2-Byte-Daten (Adressen) müssen mit einer vierstelligen Hexadezimalzahl spezifiziert werden. Eine "0" in der ersten Ziffer darf nicht ausgelassen werden.
* Bei der Eingabe über die festgelegte Anzahl Buchstaben hinausgehende Zeichen werden ignoriert.
* Mit den Monitorbefehlen ist Zugriff auf den gesamten Speicherraum möglich. Dabei ist jedoch zu beachten, daß das Vorhandensein eines einzigen Fehlers im Systemprogramm schon genügen kann, alle im MZ-800 befindlichen Daten zu zerstören.

8.2 ROM-Monitor und BASIC-Monitor

Der MZ-800 ist mit zwei Arten von Monitor ausgestattet: Der ROM-Monitor und der BASIC-Monitor. Der ROM-Monitor ist im ROM untergebracht und der BASIC-Monitors wird beim Laden des BASIC-Interpreters im RAM-Speicher abgelegt.
Der Unterschied zwischen ROM- und BASIC-Monitor ist nachstehend aufgeführt.



8.3 Starten des ROM-Monitors

Durch Einschalten des MZ-800 erscheint folgende Anzeige auf dem Bildschirm.

```
Make ready CMT
Please push key

C:  Cassette tape
M:  Monitor
```

Zum Starten des Monitors die [M]-Taste drücken. Daraufhin verändert sich der Bildschirm wie folgt.

```
**  MONITOR 9Z-504M  **
*
```

Das Sternsymbol (*) in der zweiten Zeile bezeichnen wir als Monitor-Bereitschaftszeichen, es fordert den Benutzer zur Eingabe eines Monitorbefehls auf.

Die Monitorbefehle werden im Abschnitt 8.4 erklärt.

8.4 Monitorbefehle

L-Befehl

Format L

Erklärung Dieser Befehl lädt ein Maschinenspracheprogramm von der Cassette. Wenn " ⬇ PLAY" auf dem Bildschirm angezeigt wird, die Wiedergabetaste (PLAY) drücken.

Beispiel Das folgende Beispiel lädt ein Maschinenspracheprogramm.

```
*L
⬇ PLAY ← Die [PLAY]-Taste des Cassettenrecorders drücken.
```

S-Befehl

Format S

Erklärung Dieser Befehl speichert den spezifizierten Speicherblock unter dem spezifizierten Dateinamen auf Cassette.

Beispiel Im folgenden Beispiel wird das zwischen den Adressen $6000 bis $60A3 stehende Maschinenspracheprogramm unter dem Dateinamen "MFILE" auf Cassette gespeichert. $6050 ist die Adresse, ab der das Programm auszuführen ist.

```
*S [CR]
Filename? MFILE [CR]
Top adrs? 6000 [CR]
End adrs? 60A3 [CR]
Exc adrs? 6050 [CR]
⬇ RECORD.PLAY
```

Die Aufnahmetaste (RECORD) drücken.

M-Befehl

Format	**M** <Anfangsadresse>
Erklärung	Dieser Befehl dient zur Veränderung des Speicherinhalts ab der spezifizierten Adresse.
Beispiel	Mit dem folgenden Beispiel werden die Adressen $C000 bis $C002 mit dem Ausdruck $FF und die Adressen $C010 bis $C013 mit dem Ausdruck $88 aufgefüllt.

```
*MC000 [CR]
C000 00 FF
C001 00 FF
C002 00 FF
C003 00 [SHIFT] + [BREAK]
*MC010 [CR]
C010 00 88
C011 00 88
C012 00 88
C013 00 88
C014 00 [SHIFT] + [BREAK].
```

Für die Rückkehr zum Monitor-Bereitschaftszeichen drückt man [SHIFT] + [BREAK].

J-Befehl

Format	**J**<Adresse>
Erklärung	Dieser Befehl überträgt die Ablaufsteuerung an die spezifizierte Adresse. Die <Adresse> befindet sich im Programmzähler der CPU.
Beispiel	Im folgenden Beispiel wird die Ablaufsteuerung an die Adresse $1200 übertragen.

```
*J1200 [CR]
```

G-Befehl

Format	**G** <Adresse>
Erklärung:	Dieser Befehl ruft die spezifizierte Adresse auf.
Beispiel	Im folgenden Beispiel wird die Adresse $1200 aufgerufen.

```
*G1200 [CR]
```

D-Befehl

Format

D <Anfangsadresse> <Endadresse>

Erklärung

Dieser Befehl produziert einen Speicherauszug (dump) vom spezifizierten Teil des Speichers.
Wenn die Endadresse ausgelassen wird, werden 160 Byte der Aufangsadresse angezeigt.
Das Anzeigenformat ist wie folgt:

Zu Änderung des Speicherinhalts, den Cursor zu den zu ändernden Daten bewegen, die neuen Daten eintippen und die [CR] Taste drücken.

Hinweis:
Die letzten acht Zeichen bezeichnen die ASCII Code, die den acht Hexadezimalzahlen entsprechen. Ein Steuercode ist durch einen Punkt (.) gekennzeichnet. Für den Stop der Bildschirmanzeige die Leertaste drücken. Für sofortige Rückkehr zum Monitor [BREAK] drücken, während die [SHIFT] Taste gedrückt gehalten wird.

Beispiel

Im folgenden Beispiel erfolgt ein Speicherauszug vom Inhalt der Adressen $C000 bis $C700.

*DC000C7000 [CR]

V-Befehl

Format

V

Erklärung

Dieser Befehl dient zum Verifizieren der auf Cassette gespeicherten Daten oder zur Überprüfung, ob die auf Cassette gespeicherten Daten mit den im Speicher stehenden identisch sind.
Wenn keine Datenfehler festgestellt werden, erfolgt Anzeige der Meldung "OK!" anderenfalls erfolgt Anzeige der Meldung "CHECK SUM ERROR".

Beispiel

Im folgenden Beispiel werden die Daten der Datei "MFILE" verifiziert, die mit dem S-Befehl gespeichert wurden.

*VMFILE [CR]
↓PLAY ← Die Wiedergabetaste (PLAY) drücken.

B-Befehl

Format	**B**
Erklärung	Mit diesem Befehl wird spezifiziert, daß mit jedem Drücken einer Taste der Summer ertönt. Wenn der B-Befehl nochmals eingegeben wird, stoppt der Summer.
Beispiel	* B [CR]

8.5 BASIC-Monitor

Bei Verwendung des BASIC-Interpreters wird anstelle des ROM-Monitors der BASIC-Monitor verwendet. Zum Aufrufen des BASIC-Monitors gibt man den BASIC-Befehl BYE ein.
Nachdem der BYE-Befehl eingegeben ist, wird das Bereitschaftszeichen "*" angezeigt. Nun kann man im Anschluß an das Bereitschaftszeichen einen BASIC-Monitor-Befehl eingeben.
Der BASIC-Monitor verwendet den Speicherbereich $FF00 bis $FFFF als Stack-Bereich.

8.6 BASIC-Monitor-Befehle

P-Befehl (Druck-Programmschalter)

Format: **P**

Erklärung: Dieser Befehl dient zur Ausgabe der durch den B oder F-Befehl erhaltenen Daten an den Drucker oder an den Bildschirm, je nachdem, ob zur Zeit die Drucker-Betriebsart oder die Bildschirm-Betriebsart vorgegeben ist. Beim Starten des BASIC-Monitors wird die Bildschirm-Betriebsart aktiviert, mit jeder Eingabe des P-Befehls wird diese Betriebsart dann umgeschaltet.
Wenn kein Drucker angeschlossen oder der Drucker "offline" ist erfolgt Anzeige der Fehlermeldung "ERR?" und des Monitor-Bereitschaftszeichens (*).
In diesem Falle den Drucker überprüfen und zum Aktivieren der Bildschirm-Betriebsart den P-Befehl nochmals eingeben.
Wenn der BASIC Monitor abgeufen wird, werden die beim BASIC programm vervendeten Veränderlichen micht geändert. Sie können jedock mittels Monitorbefehlen geändert werden.

D-Befehl (Dump)

Format: **D**<Anfangsadresse><Endadresse>

Erklärung: Dieser Befehl dient zur Anzeige des Inhalts vom Hauptspeicher. Wenn die Endadresse ausgelassen wird, so werden 128 Bytes ab der Anfangsadresse angezeigt. Wenn man die Anfangsadresse ausläßt, werden die nächsten 128 Bytes angezeigt.
Das Anzeigeformat ist wie folgt:

Um den Speicherinhalt zu verändern, bewegt man den Cursor zu den zu verändernden Daten, überschreibt diese mit den neuen Daten und drückt die [CR]-Taste. Daten können als 2-stellige Hexadezimalzahlen oder; (Zeichen) bestimmt werden.

Hinweis:
Die letzten 8 Zeichen zeigen 8 Hexadezimalzahlen im entsprechenden ASCII-Code an. Ein Steuercode wird durch einen Punkt (.) dargestellt. Zum Stoppen der Bildschirmanzeige drückt man die [BREAK]-Taste; für die Rückkehr zum Monitor-Bereitschaftszeichen betätigt man die [BREAK]-Taste, während man die [SHIFT]-Taste gedrückt hält.

M-Befehl (Memory set)

Format: M<Anfangsadresse>

Erklärung: Dieser Befehl verändert den Inhalt des Hauptspeichers. Wenn man die Angabe der Anfangsadresse ausläßt, erfolgt die Veränderung des Speicherinhalts ab der durch den gegenwärtigen Zeiger ausgewiesenen Adresse. Zur Rückkehr zum Monitor-Bereitschaftszeichen betätigt man die [BREAK]-Taste, während man die [SHIFT]-Taste gedrückt hält.
Durch Eingabe des M-Befehls wird der Cursor an den Daten der spezifizierten Adresse positioniert. Nun kann man die neuen Daten spezifizieren und die [CR]-Taste gemäß dem Monitor-Befehlsformat drücken. Der Adressenzeiger nimmt jeweils der Anzahl der spezifizierten Daten entsprechend zu.
Daten können als 2-stellige Hexadezimalzahlen oder; (Zeichen) bestimmt werden.

F-Befehl (Find)

Format: F<Anfangsadresse><Endadresse><Daten><Daten>...

Erklärung: Dieser Befehl sucht ein oder mehrere Daten in den spezifizierten Adressen und zeigt, wenn diese aufgefunden, die Adressen und Daten im mit dem D-Befehl gezeigten Format an. Zur Rückkehr zum Monitor-Bereitschaftszeichen betätigt man die [BREAK]-Taste während man die [SHIFT]-Taste gedrückt hält.

G-Befehl (Gosub)

Format: G<Aufrufadresse>

Erklärung: Dieser Befehl ruft die spezifizierte Adresse auf. Der Stackpointer befindet sich an der Adresse $FEFF.

T-Befehl (Transfer)

Format: T<Anfangsadresse><Endadresse><Zieladresse>

Erklärung: Dieser Befehl überträgt die Daten von der spezifizierten Ursprungsadresse zur spezifizierten Zieladresse.

S-Befehl (Save)

Format

S<Anfangsadresse><Endadresse><Ausführungsadresse>:<Gerätename>:<Dateiname>

Erklärung

Dieser Befehl speichert die in den spezifizierten Adressen stehenden Daten auf Cassette. Die Ausführungsadresse ist die Adresse, an die die Ablaufsteuerung übertragen wird, wenn das Programm mit dem L-Befehl geladen wurde. Der Dateiname wird nach einem Doppelpunkt (:) spezifiziert.

L-Befehl (Load)

Format

L<Anfangsadresse>: <Gerätename>: <Dateiname>

Erklärung

Dieser Befehl lädt die spezifizierte Datei von der Cassette. Wenn man die Anfangsadresse ausläßt, erfolgt das Laden ab der beim vorherigen Speichern der Datei mit dem S-Befehl spezifizierten Adresse. Wenn die Angabe des Dateinamens ausgelassen wird, so wird die erste aufgefundene Datei geladen. Durch Drücken der [SHIFT] + [BREAK]-Taste oder bei Auftreten eines Prüfsummenfehlers während des Ladebetriebs wird das Monitor-Bereitschaftszeichen im Anschluß an die Fehlermeldung "ERR?" angezeigt.

V-Befehl (Verify)

Format

V<Dateiname>

Erklärung

Dieser Befehl lädt die spezifizierte Datei von der Cassette und vergleicht sie mit dem im Hauptspeicher stehenden Programm bzw. Daten. Zweck dieses Befehls ist es zu überprüfen, ob das Speichern der Datei auf Cassette ordnungsgemäß ausgeführt wurde.
Wenn ein Fehler festgestellt wird, erfolgt Anzeige der Fehlermeldung "ERR?".

R-Befehl (Return)

Format

R

Erklärung

Dieser Befehl bewirkt die Rückkehr der Ablaufsteuerung zu dem Programm, von dem aus der Monitor aufgerufen wurde. Wenn der Stackpointer für das Programm, das den Aufruf des Monitors vornahm, sich an den Adressen von $FF00 bis $FFFF befindet oder wenn der Stack keine Rückspungadresse enthält, ist eine Rückführung der Ablaufsteuerung mit dem R-Befehl nicht möglich. In diesem Falle einen Warmstart mit dem G-Befehl ausführen.

Kapitel 9 Betriebsart MZ-700

9.1 Verwendung von Programmen für den MZ-700

Die meisten für Computer der Serie SHARP MZ-700 geschriebenen Programme sind auf den Computern der Serie MZ-800 ablauffähig. Bestimmte Programme, in denen der Joystick MZ-1X03 benutzt wird, können jedoch nicht verwendet werden. Für weitere Informationen darüber, ob MZ-700 Programme mit dem MZ-800 verwendet werden können, wendet man sich an den Fachhändler.
Zur Verwendung eines Programms für den MZ-700 mit dem MZ-800 muß am Gerät die MZ-700 Betriebsart aktiviert werden. Die MZ-700 Betriebsart wird eingestellt, indem man vor Betätigung des Netzschalters den Schalter 1 des Systemschalters an der Rückseite des Gerätes auf ON stellt.

MZ-700 BASIC (1Z-013) ist am Anfang der seite der mit "BASIC 1Z-013" beschrifteten Seite aufgenommen.

Drei BASIC Demonstrationsprogramme für den MZ-700 sind auf dem Band, nach dem MZ-700 BASIC aufgenommen.

Diese Programme können wie folgt ausgeführt werden.

Nach dem Laden von BASIC (1Z-013) spult man das Band zu einem der nachstehend aufgeführten Werte vor und gibt dann folgendes ein.

"OPENING" 130
"MUSIC"..... 170
"COLOR PLOTTER" 190

RUN "CMT:" [CR]

Wenn " ↓ PLAY" angezeigt wird, drückt man die [PLAY] -Taste. Nach Stoppen des Bandlauf die [STOP] -Taste drücken. Zum Stoppen des Programms die [SHIFT] und [BREAK] -Tasten gleichzeitig drücken.

9.2 Zusammenfassung der MZ-700 BASIC-Befehle und Anweisungen, Funktionen und Operationen.

Befehle

LOAD	LOAD "ABC"	Lädt die BASIC-Textdatei namens ABC von der Cassette in den Speicher.
SAVE	SAVE "E"	Speichert die gegenwärtig im Textbereich stehende BASIC-Datei unter den Namen "E" auf Cassette.
RUN	RUN	Führt das gegenwärtig im Textbereich stehende BASIC-Programm von Anfang an aus. **Hinweis:** Durch den RUN-Befehl werden alle Variablen unmittelbar vor der Programmausführung 0 bzw. null.
	RUN 1000	Das Programm wird ab Zeile 1000 ausgeführt.
MERGE	MERGE "TEST"	Liest das Programm "TEST" zusätzlich zu dem bereits im Speicher stehenden Programm ein.
VERIFY	VERIFY "H"	Zur Überprüfung, ob das gegenwärtig im BASIC-Textbereich stehende Programm und die unter dem Dateinamen "H" spezifizierte Datei auf Cassette identisch sind.
AUTO	AUTO	Generiert während des Schreibens eines Programms automatisch die Zeilennummern 10, 20, 30...
	AUTO 200,20	Generiert während des Schreibens eines Programms automatisch die Zeilennummern 200, 220, 240... in Schritten von 20 ab Zeilennummer 200. Der AUTO-Befehl wird aufgehoben, indem man [SHIFT] + [BREAK] drückt.
LIST	LIST	Zum Auflisten des gesamten im BASIC-Textbereich stehenden Programms auf dem Bildschirm.
	LIST – 500	Anzeige des Programm-Listings bis Zeilennummer 500.
LIST/P	LIST/P	Das Listing des Programms erfolgt über den Drucker in der TEXT-Betriebsart.
RENUM	RENUM	Zur Neunumerierung der Programmzeilen.
	RENUM 100	Zur Neunumerierung der Programmzeilen in Zehnerschritten ab Zeilennummer 100.
NEW	NEW	Löscht das gegenwärtig im Textbereich stehende BASIC-Programm und den Variablenbereich. Der durch den LIMIT-Befehl vorgegebene Maschinensprachebereich wird nicht gelöscht.
CONT	CONT	Zur Fortsetzung der Programmausführung. Mit anderen Worten wird die Ausführung ab der durch Drücken der [SHIFT] + [BREAK]-Tasten oder der STOP-Anweisung im Programm vorgenommenen Unterbrechung fortgesetzt. Der CONT-Befehl wird ungültig, wenn man während einer Unterbrechung der Programmausführung am BASIC-Text Veränderungen vornimmt.
BYE	BYE	Überträgt die Systemsteuerung von BASIC an das Monitorprogramm. (Die Rückkehr vom Monitor zum BASIC kann durch Ausführung des Monitor-Befehls "R" vorgenommen werden.
KEY LIST	KEY LIST	Zum Auflisten des Inhalts, mit dem die definierbaren Funktionstasten zur Zeit belegt sind.

Dateisteuerbefehle

WOPEN	WOPEN "DATA"	Eröffnet eine Datendatei auf Cassette, um daraufhin Daten in diese Datei zu schreiben. In unserem Beispiel wird der Datendatei der Name "DATA" zugewiesen.
PRINT/T	PRINT/T X	Schreibt die Daten in demselben Format auf Cassette, in dem die Anzeige durch den PRINT-Befehl erfolgen würde.
ROPEN	ROPEN "DATA"	Sucht die Datendatei DATA auf Cassette und öffnet diese Datei, um das Lesen von Daten daraus vorzubereiten.
INPUT/T	INPUT/T X	Einlesen der Daten von einer Cassette und Zuweisung dieser Daten zur Variablen X.
CLOSE	CLOSE	Zum Schließen der Cassetten-Datendateien, nachdem der Schreib- bzw. Lesebetrieb abgeschlossen ist.

Fehlerverarbeitungs-Anweisungen

ON ERROR GOTO	ON ERROR GOTO 1000	Die Anweisung besagt, daß bei Auftreten eines Fehlers während der Programmausführung die Ablaufsteuerung an Zeile 1000 übertragen werden soll.
IF ERN	IF ERN=43 THEN 1050	Wenn die Fehlernummer 43 beträgt, wird mit dieser Anweisung die Ablaufsteuerung an Zeile 1050 übertragen.
IF ERL	IF ERL=350 THEN 1090	Wenn der Fehler in Zeile 350 auftrat, wird mit dieser Anweisung die Ablaufsteuerung an Zeile 1090 übertragen.
	IF (ERN=43)* (ERL=700) THEN END	Durch diese Anweisung wird die Programmausführung beendet, wenn die Fehlernummer 43 war und der Fehler in Zeilen 700 auftrat. Wenn bei BASIC ein Fehler während der Programmausführung auftritt, werden Fehlernummer und Fehlerzeilennummer die Variablen ERN und ERL zugewiesen.
RESUME	650 RESUME	Überträgt die Ablaufsteuerung an den Befehl, der den Fehler verursachte.
	700 RESUME NEXT	Überträgt die Ablaufsteuerung an den nächsten Befehl nach dem den Fehler verursachenden Befehl.
	750 RESUME 400	Überträgt die Ablaufsteuerung an Zeile 400.
	800 RESUME 0	Überträgt die Ablaufsteuerung an den Programmanfang.

Substitutions-Anweisungen

LET	LET A=X+3	Setzt die Summe aus der numerischen Variable X und dem numerischen Wert 3 mit der numerischen Variable A gleich. LET kann auch ausgelassen werden.

Eingabe/Ausgabe- und Farbsteuer-Anweisungen

COLOR	10 COLOR,,,2	Die gesamte Bildschirm-Hintergrundfarbe wird auf rot umgeschaltet.
	20 COLOR 3,2,7	Die Farbe der Zeichen an den Koordinaten (3,2) wird auf weiß umgeschaltet.
	30 COLOR 4,2,4,2	Die Farbe der Zeichen an den Koordinaten (4,2) wird grün und die Hintergrundfarbe rot.
PRINT	10 PRINT A	Zeigt den Inhalt der numerischen Variable A auf dem Bildschirm an.
	?A$	Zeigt den Inhalt der Stringvariable A$ auf dem Bildschirm an.
	100 PRINT [6,5] "ABC"	Schreibt die Zeichenfolge "ABC" in gelb auf hellblauem Hintergrund.
	110 PRINT [,4] "DEF"	Schreibt die Zeichenfolge "DEF" in gelb auf grünem Hintergrund.
	120 PRINT [7,4] "GHI"	Schreibt die Zeichenfolge "GHI" in weiß auf grünem Hintergrund.
	200 PRINT	Falls nur PRINT angewiesen ist, erfolgt Zeilenwechsel.
PRINT USING	PRINT USING "# # #.# #";A	Gibt ein Format vor, mit der die Anzahl der Dezimalstellen durch eine feste Komma-Anzeige vorgegeben wird.
INPUT	10 INPUT A	Zur Eingabe eines Wertes über die Tastatur, der der Variablen A zugewiesen wird.
	20 INPUT A$	Zur Eingabe einer Zeichenfolge über die Tastatur, der der Stringvariablen A$ zugewiesen wird.
	30 INPUT "VALUE?";D	Vor der Eingabe über die Tastatur wird die Meldung (Frage) "VALUE?" angezeigt. Das Semikolon dient zur Trennung der Bildschirmmeldung-Zeichenfolge von der Variablen.
	40 INPUT X, X$, Y, Y$	Numerische Variablen und Stringvariablen können unter Verwendung des Kommas miteinander kombiniert werden. Bei der Eingabe ist jedoch darauf zu achten, daß die Variablentypen jeweils übereinstimmen.
SET	SET 30,15	Setzt die Position der Koordinaten (30, 15) zu.
RESET	RESET 30,15	Löscht die Position der führenden Koordinaten (30, 15).
GET	10 GET N	Zur Eingabe eines numerischen Zeichens über die Tastatur, daß der numerischen Variablen N zugewiesen wird. Wenn zu diesem Zeitpunkt keine Taste betätigt wird, erfolgt Eingabe von 0.
	20 GET K$	Zur Eingabe einer Zeichenfolge über die Tastatur, die der Stringvariablen K$ zugewiesen wird. Erfolgt zu diesem Zeitpunkt keine Tastenbetätigung, bleibt K$ leer.
READ~DATA	10 READ A,B,C 1010 DATA 25, −0.5, 500	Die numerischen Daten 25, −0.5 und 500 werden durch Ausführung der links stehenden READ-DATA-Anweisungen jeweils den numerischen Variablen A, B und C zugewiesen.
	10 READ H$,H, S$,S 30 DATA "HEART", 3 35 DATA "SPADE", 11	Die ersten Daten der DATA-Anweisung, d.h die Stringdaten "HEART" werden der ersten Variable der READ-Anweisung zugewiesen, d.h. der Stringvariable H$. Daraufhin werden die numerischen Daten 3 der zweiten Variable H zugewiesen, woraufhin der Lesebetrieb hintereinander fortgesetzt wird.

RESTORE	10 READ A,B,C 20 RESTORE 30 READ D,E 100 DATA 3, 6, 9, 12, 15	Im links stehenden Beispiel werden die numerischen Werte 3, 6 und 9 durch die READ-Anweisung in Zeile 10 den Variablen A, B und C zugewiesen, da jedoch als nächstes die RESTORE-Anweisung auftritt, werden den Variablen D und E in der READ-Anweisung von Zeile 30 nicht 12 und 15, sondern wiederum 3 und 6 zugewiesen.
	700 RESTORE 200	Bewegt den Zeiger zum Lesen von Daten in der READ-DATA-Anweisung an den Anfang der DATA-Anweisung in Zeile 200.

Schleifen-Anweisung

FOR ~ NEXT	10 FOR A = 1 TO 10 20 PRINT A 30 NEXT A	Zeile 10 enthält einen Befehl zur Veränderung der Variablen A, indem ihr nacheinander Werte von 1 bis 10 zugewiesen werden, der Wert des ersten A wird 1. Zeile 20 bewirkt die Anzeige von A auf dem Bildschirm, es wird daher die Ziffer 1 angezeigt. Daraufhin wird A in Zeile 30 um 1 erhöht, wird also 2 und die gesamte Schleife mehrmals wiederholt, bis der Wert von A 10 wird. Wenn die Schleife endet, wird der A zugewiesene Wert 11.
	10 FOR B = 2 TO 8 STEP 3 20 PRINT B 30 NEXT B	Befehl zur Veränderung der Variablen B, der die Werte von 2 bis 8 mit einer Zunahme (Schritt) von 3 zugewiesen werden (in Zeile 10). Der STEP-Wert kann auch negativ sein, so daß die Variable mit jedem Schleifendurchgang kleiner wird.
	A-Schleife: 10 FOR A = 1 TO 3 B-Schleife: 20 FOR B = 10 TO 30 B-Schleife: 30 PRINT A, B B-Schleife: 40 NEXT B A-Schleife: 50 NEXT A	Beispiel für eine Verschachtelung von FOR-NEXT-Schleifen (Variablen A und B zu). Bitte beachten, daß die Schleife B vollständig in Schleife A enthalten sein muß. Schleifen können auch mehrmals ineinander verschachtelt werden, nur muß die innere Schleife sich jeweils vollständig innerhalb der äußeren Schleife befinden.

Verzweigungs-Anweisungen

GOTO	100 GOTO 200	Es erfolgt ein Sprung in der Programmausführung zur Zeile 200.
GOSUB ~ RETURN	100 GOSUB 700 ..... 800 RETURN	Verzweigt die Programmausführung zum Unterprogramm in Zeile 700 (Aufruf des Unterprogramms). Die Unterprogramm-Ausführung wird durch die RETURN-Anweisung beendet, woraufhin die auf den GOSUB-Befehl folgende Anweisung im Hauptprogramm ausgeführt wird.
IF ~ THEN	10 IF A > 20 THEN 200	Wenn die Variable A größer als 20 ist, erfolgt Sprung zur Zeile 200. Wenn A kleiner oder gleich 20 ist, wird die auf diese Zeile folgende Anweisung ausgeführt.
	50 IF B < 3 THEN B = B + 3	Wenn die Variable B kleiner als 3 ist, so erhält sie den neuen Wert B + 3. Wenn B größer oder gleich 3 ist, wird die nächste Anweisung ausgeführt.

IF ~ GOTO	100 IF A > = B GOTO 10	Wenn die Variable A größer oder gleich der Variablen B ist, erfolgt Sprung zur Zeile 10. Wenn A kleiner als B ist, wird die nächste Anweisung ausgeführt.
IF ~ GOSUB	30 IF A = B * 2 GOSUB 90	Wenn der Wert der Variablen A dem doppelten Wert von B gleich ist, erfolgt Verzweigung der Programmausführung zum Unterprogramm in Zeile 90. Andernfalls wird die nächste Anweisung ausgeführt. Wenn eine ON-Anweisung von mehreren Anweisungen gefolgt wird, erfolgt Ausführung dieser Anweisungen, wenn diese Bedingung nicht erfüllt wird; ist es jedoch eine IF-Anweisung, wird die Ausführung mit der nächsten Zeilennummer fortgesetzt, wenn die Bedingung nicht erfüllt ist, d.h. in diesem Falle werden die auf IF folgenden mehrfachen Anweisungen ignoriert.
ON ~ GOTO	50 ON A GOTO 70, 80, 90	Sprung der Programmausführung nach Zeile 70, wenn Variable A gleich 1, nach Zeile 80 wenn A gleich 2 und nach Zeile 90 wenn Variable A gleich 3 ist. Wenn A 0 bzw. 4 oder größer ist, wird die auf diese Zeile folgende Anweisung ausgeführt. ON enthält die INT-Funktion, wenn daher z.B. A = 2.7, erfolgt die Ausführung in der gleichen Weise, wie wenn A gleich 2 ist, d.h. nach Zeile 80.
ON ~ GOSUB	90 ON A GOSUB 700, 800	Wenn Variable A gleich 1, erfolgt Verzweigung zum Unterprogramm in Zeile 700, und wenn A gleich 2, wird zum Unterprogramm in Zeile 800 verzweigt. Wenn A gleich 0 bzw. 3 oder mehr ist, erfolgt Ausführung der auf diese Zeile folgenden Anweisung.

Definition-Anweisungen

DIM	10 DIM A (20)	Für die eindimensionale numerische Feldvariable A ( ) stehen 21 Elemente zur Verfügung, von A (0) bis A (20).
	20 DIM B (79,79)	Für die zweidimensionale numerische Feldvariable B ( ) stehen 6400 Elemente zur Verfügung, und zwar von B (0) bis B (79, 79).
	30 DIM C1$ (10)	Für die eindimensionale Stringfeldvariable C1$ ( ) stehen 11 Elemente zur Verfügung, von C1$ (0) bis C1$ (10).
	40 DIM K$ (7,5)	Für die zweidimensionale Stringfeldvariable K$ ( ) stehen 48 Elemente zur Verfügung, von K$ (0) bis K$ (7,5).
DEF FN	100 DEF FNA (X) = X↑2 − X 110 DEF FNB (X) = LOG (X) + 1 120 DEF FNZ (Y) = LN (Y)	In Zeile 100 wird X$ − X als FNA (X) definiert, in Zeile 110 wird $\log_{10}X + 1$ als FNB (X) definiert und in Zeile 120 wird $\log_e$ Y als FNZ(Y) definiert.
DEF KEY	15 DEF KEY (1) = "LIST" + CHR$ (13) 25 DEF KEY (2) = "LOAD:RUN" + CHR$ (13)	Jede Funktion ist auf eine Variable begrenzt. Die DEF KEY-Anweisung in Zeile 15 weist der Funktionstaste Nr. 1 die Funktion LIST [CR] zu, und Zeile 25 weist die Funktion LOAD:RUN [CR] der Funktionstaste Nr. 2 zu.

Kommentar- und Steuerbefehle

REM	200 REM JOB1	REM ist eine Kommentar-Anweisung, sie wird bei der Programmausführung ignoriert.
STOP	850 STOP	Stoppt die Programmausführung und wartet auf einen Direktbefehl. Wenn nun CONT eingegeben wird, erfolgt Fortsetzung der Programmausführung.
CLR	300 CLR	Alle numerischen Variablen und Zeichenvariablen werden 0 oder leer (null). Alle Feldvariablen-Definitionen werden aufgehoben. Desgleichen werden alle DEF FN-Anweisungen ungültig.
CURSOR	50 CURSOR 25, 15 60 PRINT "ABC"	Spezifiziert die Position durch Zahlen oder Variablen: von 0 bis 39, vom linken Ende in Richtung X-Achse und von 0 bis 24 vom oberen Ende in Richtung Y-Achse. Im links stehenden Beispiel wird die Zeichenfolge "ABC" von der 26-ten Cursorposition ab dem linken Ende des Bildschirms und der 16-ten Cursorposition von oben angezeigt.
CONSOLE	10 CONSOLE 0, 25, 0, 40	Der Scroll-Bereich umfaßt den gesamten Bildschirm.
	20 CONSOLE 5, 15	Spezifiziert den Scroll-Bereich von der 5-ten bis zur 15-ten Zeile.
	30 CONSOLE 0, 25, 5, 30	Spezifiziert den Scroll-Bereich von der 5-ten bis zur 30-ten Zeile.
	40 CONSOLE 0, 10, 0, 10	Spezifiziert den Scroll-Bereich im Bereich 10×10.
SIZE	? SIZE	Zeigt die Größe des unbenutzten BASIC-Textbereiches (in Bytes) an.
TI$	100 TI$ = "222030"	Stellt die interne Uhr auf 10:20:30 p.m. ein. Die Zeitdaten werden als 6-stellige Zahl innerhalb von Fragezeichen ausgedrückt.

Musik-Steuerbefehle

MUSIC TEMPO	300 TEMPO 7 310 MUSIC"DE# FGA"	Tempo 7 (schnellstes Tempo) wird durch die Anweisung in Zeile 200 spezifiziert. Durch Zeile 310 werden die Töne re, mi, fa# so la (Mittelbereich) im Tempo 7 abgespielt. Wenn keine TEMPO-Anweisung erfolgt, wird die Musik dem Tempo der Standardvorgabe entsprechend abgespielt.
	300 M1$ = "C3E G+C" 310 M2$ = "BGD −G" 320 M3$ = "C85R" 330 MUSIC M1$, M2$,M3$	In diesem Beispiel wird die Melodie den 3 Stringvariablen zugewiesen und der MUSIC-Befehl ausgeführt. Bei Verwendung der geschriebenen Noten werden die nachstehenden Töne abgespielt. Bitte beachten, daß der Abspielbetrieb im Stadardvorgabetempo erfolgt, da keine TEMPO-Anweisung vorgegeben ist.

Befehle zur Ablaufsteuerung von Maschinensprache-Programmen.

INP@	INP@$E8,A	Weist die Daten an Port Nr. $E8 der Variablen A zu.
OUT@	OUT@$E8,A	Ausgabe von Variable A an Port Nr. $E8.
LIMIT	100 LIMIT 49151	Begrenzt den vom BASIC-Programm verwendeten Bereich auf die Adresse 49151 (BFFF in hexadezimaler Schreibweise).
	100 LIMIT A	Begrenzt den vom BASIC-Programm verwendeten Bereich auf die in der Variablen A enthaltene Adresse.
	100 LIMIT $BFFF	Begrenzt den vom BASIC-Programm verwendeten Bereich auf die Adresse auf BFFF in hexadezimaler Schreibweise. Hexadezimale Schreibweise wird durch ein der Zahl vorgestelltes "$" Zeichen ausgewiesen.
	300 LIMIT MAX	Wiederherstellung des Maximums für den vom BASIC-Programm verwendbaren Bereich.
POKE	120 POKE 49450, 175	Belegt die Adresse 49450 (dezimale Schreibweise) mit den Daten 175 (dezimale Schreibweise).
	130 POKE AD, DA	Belegt die durch die Variable AD spezifizierte Adresse mit dem durch die Variable DA ausgewiesenen Wert (0 bis 255).
PEEK	150 A=PEEK (49450)	Wandelt die Daten an der Adresse 49450 (Adresse in Dezimalschreibweise) in eine Dezimalzahl um und weist sie der Variablen A zu.
	160 B=PEEK (C)	Wandelt die an der durch Variable C spezifizierten Adresse in Dezimalschreibweise vorhandenen Daten in eine Dezimalzahl um und weist sie der Variablen B zu.
USR	500 USR (49152)	Überträgt die Ablaufsteuerung an die Dezimaladresse 49152. Diese Steueranweisung hat die gleiche Funktion wie der Maschinensprache-Befehl CALL. Daher erfolgt bei Ausführung eines RET-Befehls im Maschinensprache-Programm (201 in Dezimalschreibweise) Rückkehr zum BASIC-Programm.
	550 USR (AD)	Ruft die durch die Variable AD spezifizierte Dezimaladresse auf.
	570 USR ($C000)	Ruft die Hexadezimal-Adresse C000.

Drucker-Steuerbefehle

AXIS		Gilt in der GRAPH-Betriebsart.
	30 AXIS 0, −10, 48	Versieht die Y-Koordinaten-Achse ab der gegenwärtigen Stiftposition mit 48 Skalenunterteilungen mit einer Zunahme von je 10.
	50 AXIS 1, 10, 48	Versieht die X-Koordinaten-Achse ab der gegenwärtigen Stiftposition mit 48 Skalenunterteilungen mit einer Zunahme von jeweils 10.
CIRCLE		Gilt in der GRAPH-Betriebsart.
	50 CIRCLE 0, 0, 240, 0, 360, 0, 2	Zeichnet einen Kreis (Radius 240) mit den Koordinaten (0,0) als Mittelpunkt.

GPRINT		Gilt in der GRAPH-Betriebsart.
	30 GPRINT (2,2), "A"	Druckt das Zeichen A auf dem Kopf stehend im 26-Stellen-Modus der TEXT-Betriebsart aus.
HSET	HSET	Spezifiziert die gegenwärtige Stiftposition als den neuen Start. (Gilt in der GRAPH-Betriebsart)
LINE		Gilt in der GRAPH-Betriebsart
	10 LINE% 1, 240, 0, 240, −240, 0, −240, 0, 0	Die Koordinaten (240,0) (240, −240) (0, −240) und (0,0) werden mit einer durchgehenden Linie ab der gegenwärtigen Stiftposition verbunden.
MODE	MODE TN	Rückkehr von der GRAPH-Betriebsart zur TEXT-Betriebsart (40 Zeichen pro Zeile).
	MODE TL	Rückkehr von der GRAPH-Betriebsart zur TEXT-Betriebsart (26 Zeichen pro Zeile).
	MODE TS	Rückkehr von der GRAPH-Betriebsart zur TEXT-Betriebsart (80 Zeichen pro Zeile).
	MODE GR	Umschaltung von der TEXT-Betriebsart auf die GRAPH-Betriebsart (zum Zeichnen von Grafik und Umrißzeichnungen).
MOVE		Gilt in der GRAPH-Betriebsart.
	10 MOVE 150, 100	Bewegt den Stift auf der gegenwärtigen Stiftposition aufwärts zu den Koordinaten (150, 100).
RMOVE		Gilt in der GRAPH-Betriebsart.
	20 RMOVE −240, 240	Bewegt den Stift bezogen auf die gegenwärtige Stiftposition aufwärts, um −240, X-Richtung und 240 (Y-Richtung).
PAGE		Gilt in der TEXT-Betriebsart
	10 PAGE 30	Spezifiziert 30 Zeilen pro Seite.
PCOLOR		Gilt in der TEXT- und in der GRAPH-Betriebsart.
	10 PCOLOR 1 20 PRINT/P "ABC"	Druckt mit dem Plotter "ABC" in blau.
PHOME	PHOME	Bewegt den Stift ab der gegenwärtigen Stiftposition in Aufwärtsrichtung und bewirkt Rückkehr zum Startpunkt. (Gilt in der GRAPH-Betriebsart).
PLOT	PLOT ON	Ermöglicht die Verwendung des Farbplotters als Ersatz für die Anzeige. (Gilt in der TEXT-Betriebsart).
	PLOT OFF	Zur Aufhebung der obigen Funktion.
PRINT/P		Gilt in der TEXT-Betriebsart.
	10 PRINT/P A, A$	Ausgabe des Inhaltes der Stringvariablen A$ im Anschluß an die Ausgabe des Inhaltes der numerischen Variable A mit dem Drucker.
	20 PRINT/P "H"	Formularvorschub des Druckers.
PRINT/P USING		Zur Vorgabe des Formats, in dem die Daten über den Drucker ausgegeben werden. Die Formatspezifikation erfolgt im Anschluß an das Wort USING.
	PRINT/P USING "# # # #";A	Der Inhalt der numerischen Variable A wird mit rechten Randausgleich innerhalb von 4 Stellen an den Drucker ausgegeben.

RLINE		Gilt in der GRAPH-Betriebsart.
	70 RLINE% 1, 240, 0, −120, −SQ, −120, SQ	Verbindet die spezifizierten Positionen, bezogen auf die gegenwärtige Stiftposition (240,0), (−120, −SQ, und (−120,SQ) mit einer durchgehenden Linie.
SKIP		Gilt in der TEXT-Betriebsart.
	10 SKIP 10	Bewegt das Papier um 10 Zeilen in Vorwärtsrichtung.
	20 SKIP −10	Bewegt das Papier um 10 Zeilen in Rückwärtsrichtung.
TEST	TEST	Zur Überprüfung der Farbspezifikation, der vorhandenen Tintenmenge und Stift-Trockenheit. (Gilt in der TEXT-Betriebsart).

Arithmetische Funktionen

ABS (X)	A=ABS (X)	Weist den absoluten Wert der Variable |X| der Variablen A zu. Beispiel: A=ABS (−2.9)→A=2.9 A=ABS (−5.5)→A=5.5
SGN (X)	A=SGN (X)	Weist das Vorzeichen der Variablen X der Variablen A zu. Wenn der Wert von X negativ ist, wird A −1 zugewiesen, wenn X 0 ist wird A 0 zugewiesen und wenn X positiv ist, wird A 1 zugewiesen. A={ 1 (X>) 0 (X=0) −1 (X<) Beispiel: Durch Ausführung von A=SGN (0.4) wird der Variablen A 1 zugewiesen.
INT (X)	A=INT (X)	Weist A die größte Ganzzahlvariable zu, die kleiner oder gleich dem Wert der Variablen X ist. Beispiele: A=INIT (3.87)−→A=3 A=INIT (0.6) →A=0 A=INIT (−3.87)→A=−4
SIN (X)	A=SIN (X) A=SIN(30∗PAI(1/180)	Weist den Sinus von X (wobei X in Radian angegeben wird) der Variablen A zu. Wenn der Wert von X in Grad angegeben wird, muß man, um den Sinus zu erhalten, vor Verwendung dieser Funktion den Wert in Radian umwandeln. Da ein Grad /180 Radian entspricht, wird der Wert in Radian erhalten, indem man die Gradzahl mit PAI (1/180) multipliziert. Zum Beispiel ist 30° = 30∗PAI(1/180 Radian). Das gleiche gilt für die Funktionen COS, und ATN. Weist den Kosinus von X (wobei X in Radian angegeben wird) der Variablen A zu.
COS (X)	A=COS (X) A=COS (200∗PAI(1)/180)	Weist den Tangens von X (wobei X in Radian angegeben wird) der Variablen A zu.
TAN (X)	A=TAN (X) A=TAN(Y∗PAI(1)/180)	Weist den Tangens von X (wobei X in Radian angegeben wird) der Variablen A zu.
ATN (X)	A=ATN (X) A=180/PAI(1)∗ATN(X)	Weist den Arkustangens in Radian von X ($\tan^{-1}X$) der Variablen A zu. Der angegebene Wert liegt im Bereich von $-\pi/2$ bis $\pi/2$.
SQR (X)	A=SQR (X)	Berechnet die Quadratwurzel von X und weist das Ergebnis der Variablen A zu. X muß eine positive Zahl oder null sein.
EXP (X)	A=EXP (X)	Berechnet den Wert von e^x und weist das Ergebnis der Variablen A zu.
LOG (X)	A=LOG (X)	Berechnet den Zehnerlogarithmus von X ($\log_{10}$ X) zu und weist das Ergebnis der Variablen A zu.
LN (X)	A=LN (X)	Berechnet den natürlichen Logaritmus von X ($\log_e$ X) und weist das Ergebnis der Variablen A zu.
PAI (X)	A=PAI (X)	Weist den Wert, der X-mal dem Wert von π entspricht, der Variablen A zu.

RAD (X)	A=RAD (X)	Wandelt den Wert von X (wobei X in Grad angegeben wird) in Radian um und weist das Ergebnis der Variablen A zu.

Stringverarbeitungs-Funktionen

LEFT$	10 A$=LEFT$ (X$,N)	Setzt die Stringvariable A$ mit der Stringvariable X$ (ab Anfang bis zum N-ten Zeichen) gleich. Es spielt keine Rolle, ob N eine Konstante, Variable oder numerische Formel ist.
MID$	20 B$=MID$ (X$, M, N)	Setzt die Stringvariable B$ mit der Stringvariable X$ (ab dem M-ten Zeichen bis zum N-ten-Zeichen) gleich.
RIGHT$	30 C$=RIGHT$ (X$, N)	Setzt die Stringvariable C$ mit der Stringvariable X$ (ab dem Ende bis zum N-ten Zeichen) gleich.
SPC	40 D$=SPC (N)	Weist der Stringvariablen D$ N-Anzahl Leerstellen zu.
CHR$	60 F$=CHR$(A)	Entgegengesetzt zur ASC-Funktion, weist das dem Wert der reellen Zahl A entsprechende ASCII-Codezeichen der Stringvariable F$ zu. Dabei kann A eine Konstante, Variable oder numerische Formel sein.
ASC	70 A=ASC (X$)	Weist den Wert vom ASCII-Code des ersten Zeichens der Stringvariable X$ der Variablen A zu.
STR$	80 N$=STR$ (1)	Entgegengesetzt zur VAL-Funktion; die numerische Variable I wird als Zeichenfolge behandelt, die der Stringvariablen N$ zugewiesen wird.
VAL	90 I=VAL (N$)	Weist den numerischen Wert der Stringvariable N$ der numerischen Variable I zu.
LEN	100 LX=LEN (X$)	Weist die Länge der Zeichenfolge X$ (Anzahl der in der Zeichenfolge enthaltenen Zeichen) der Variable LX zu.
	100 LS=LEN (X$+Y$)	Weist die Summe der Längen von Stringvariable X$ und Y$ der Variablen LS zu.

Tab-Funktion

TAB	10 PRINT TAB (X);A	Zur Anzeige des Wertes der Variablen A in der Zeichenposition X+1 gerechnet ab dem linken Rand des Bildschirms.

Arithmetische Operationen

Die Rechenreihenfolge entspricht der Darstellung der weißen Zeichen auf schwarzem Untergrund links, jedoch haben Klammerrechnungen ( ) eine noch höhere Priorität.

↑	10 A=X↑Y (Potenz)	Weist das Ergebnis der Rechnung X↑Y der Variablen A zu. (Hierbei ist jedoch zu beachten, daß ein Fehler auftritt, wenn Y kein Integral ist, im Falle daß X bei X↑Y eine negetive Zahl ist.
–	10 A=–B (Minuszeichen)	0–B ist eine Subtraktion, bitte beachten, daß "–" von –B ein Minuszeichen ist.
*	10 A=X*Y (Multiplikation)	Weist das Multiplikationsergebnis von X und Y der Variablen A zu.
/	10 A=X/Y (Division)	Weist das Divisionsergebnis von X und Y der Variablen A zu.
+	10 A=X+Y (Addition)	Weist das Additionsergebnis von X und Y der Variablen A zu.

–	10 A=X–Y (Subtraktion)	Weist das Subtraktionsergebnis von X und Y der Variablen A zu.

Logische Vergleichsoperatoren

=	10 IF A=X THEN	Wenn die Variablen A und X gleich sind, werden die auf THEN folgenden Befehle ausgeführt.
	20 IF A$="XYZ" THEN.....	Wenn der Inhalt der Stringvariable A$ String XYZ ist, werden die auf THEN folgenden Befehle ausgeführt.
>	10 IF A>X THEN	Wenn die Variable A größer als X ist, werden die auf THEN folgende Befehle ausgeführt.
<	10 IF A<X THEN	Wenn die Variable A kleiner als X ist, werden die auf THEN folgenden Befehle ausgeführt.
<> **oder** ><	10 IF A<>F THEN.....	Wenn die Variablen A und X ungleich sind, werden die auf THEN folgenden Befehle ausgeführt.
>= **oder** =>	10 IF A>=X THEN.....	Wenn die Variable A größer oder gleich X ist, werden die auf THEN folgenden Befehle ausgeführt.
<= **oder** =<	10 IF A<=X THEN.....	Wenn die Variable A kleiner oder gleich X ist, werden die auf THEN folgenden Befehle ausgeführt.
*	40 IF (A>X)* (B>Y) THEN...	Wenn die Variable A größer als X ist und die Variable B größer als Y ist, werden die auf THEN folgenden Befehle ausgeführt.
+	50 IF (A>X)+ (B>Y) THEN.....	Wenn Variable A größer als X oder Variable B größer als Y ist, werden die auf THEN folgenden Befehle ausgeführt.

Weitere Symbole

?	200 ?"A+B="; A+B 210 PRINT "A+B =";A+B	Kann anstelle von PRINT verwendet werden. Demzufolge sind die Zeilen 200 und 210 gleich.
:	220 A=X:B=X↑2 :?A,B	Symbole zur Unterteilung der Programmzeile, wenn diese mehrere Anweisungen enthält. In unserem Beispiel enthält die Zeile 220 drei separate Anweisungen.
;	230 PRINT"AB"; "CD"; "EF"	Zur kontinuierlichen Ausführung von PRINT. Demzufolge wird durch Ausführung von Zeile 230 "ABCDEF" kontinuierlich hintereinander, ohne Abstand zwischen den einzelnen Zeichenfolgen angezeigt.
	240 INPUT"X="; X$	Zeigt "X=" auf dem Bildschirm an und wartet auf die Eingabe der Stringvariablen X$ über die Tastatur.
,	250 PRINT"AB", "CD","E"	Ausführung von PRINT mit Tabulator. Durch Ausführung von Zeile 250 wird AB angezeigt, daraufhin erscheint CD um 10 Zeichen von A nach rechts verschoben auf dem Bildschirm, dann wird E um 10 Zeichen nach rechts von C verschoben auf dem Bildschrim angezeigt.
	300 DIM A(20), B$(3,6)	Beispiel zur Zeichensetzung bei einer Variable B$ (3,6)
" "	330 B$= "MZ-700"	"Markiert den Inhalt einer Zeichenfolge (String)"
$	340 C$="ABC" +CHR$(3)	Kennzeichnet eine Stringvariable als solche.
	500 LIMIT $BFFF	Kennzeichnet eine Hexadezimalzahl als solche.

π	550 S=SIN (X * π/180)	Der ungefähre Wert von pi (3.141597) wird durch π ausgedrückt.

Anhang

Anhang A Anzeigensteuerung bei MZ-800 Betrieb

(1) Grafik-Speicher

In der Standardausführung unterstützt der MZ-800 einen Bildschirm mit 320 × 200 Punkten in vier Farben, die aus 16 Farben ausgewählt werden oder eine einfarbige Anzeige von 640 × 200 Punkten. Durch Einbau eines optionalen Grafik-Speicher-Chips (MZ-1R25) wird das Anzeigevermögen so verbessert, daß 320 × 200 Punkte in 16 Farben oder 640 × 200 Punkte in 4 aus 16 wählbaren Farben angezeigt werden können.

(2) 40-Zeichen-Betriebsart und 80-Zeichen-Betriebsart (Zeichenanzeige)

Die Anzahl der angezeigten Zeichen pro Zeile kann mit dem INIT-Befehl zwischen 40 und 80 umgeschaltet werden.

(3) Grafikanzeige-Betriebsarten

Die Auflösung und die Anzahl der gleichzeitig anzeigbaren Farben variiert je nach der gewählten Grafikanzeige-Betriebsart. Die folgenden Grafik-Anzeigebetriebsarten stehen zur Wahl.

Betriebs-art	Auflösung	Zeichen pro Zeile	Farben
1	320 × 200 Punkte	40	4 Farben
2	320 × 200 Punkte		16 Farben
3	640 × 200 Punkte	80	Vordergrund- und Hintergrundfarben
4	640 × 200 Punkte		4 Farben

Die Grafikanzeige-Betriebsart wird durch den Mn-Parameter des INIT-Befehls vorgegeben.

Beispiel:

INIT "CRT:M1" ..... aktiviert die Betriebsart 1.
INIT "CRT:M2" ..... aktiviert die Betriebsart 2.

(4) Farbpalette

Die gleichzeitig anzeigbaren Farben werden anhand der Farbpalette bestimmt. Die Farbpalette dient zur Wahl von 4 aus 16 Farben. Die nachstehende Aufstellung enthält die 16 Farben, die angezeigt und durch die Farbcodes gekennzeichnet werden können.

Farbcode	Farbe
0	Schwarz
1	Blau
2	Rot
3	Magenta
4	Grün
5	Cyan
6	Gelb
7	Weiß
8	Grau
9	Hellblau
10	Hellrot
11	Hellmagenta
12	Hellgrün
13	Hellcyan
14	Hellgelb
15	Hellweiß

In Betriebsart 1 oder Betriebsart 4 werden die Palettencodes von 0 bis 3 verwendet. Die Anfangseinstellungen der Farbcode-Zuweisungen für die Palettencodes ist nachstehend aufgeführt.

Palettencode	Farbcode (Farbe)
0	0 (schwarz)
1	1 (blau)
2	2 (rot)
3	15 (hellweiß)

In Betriebsart 3 werden die Palettencodes 0 und 1 verwendet und die Anfangseinstellungen der Farbcode-Zuweisungen sind wie folgt.

Palettencode	Farbcode (Farbe)
0	0 (schwarz)
1	15 (hellweiß)

Die Farbcode-Zuweisungen zu den Palettencodes können mit dem PAL-Befehl verändert werden.

Beispiel:

PAL 0,4 ..... Weist dem Palettencode 0 den Farbcode 4 (grün) zu.
PAL 2,7 ..... Weist dem Palettencode 2 den Farbcode 7 (weiß) zu.

(5) Verwendung der Palette in der Betriebsart 2

In der Betriebsart 2 sind die Anfangseinstellungen der Farbcode-Zuweisungen zu den Palettencodes wie folgt.

Palettencode	Farbcode (Farbe)
0	0 (schwarz)
1	1 (blau)
2	2 (rot)
3	3 (magenta)
4	4 (grün)
5	5 (cyan)
6	6 (gelb)
7	7 (weiß)
8	8 (grau)
9	9 (hellblau)
10	10 (hellrot)
11	11 (hellmagenta)
12	12 (hellgrün)
13	13 (hellcyan)
14	14 (hellgelb)
15	15 (hellweiß)

Die Verwendung der Palette in der Betriebsart 2 ist etwas kompliziert. In der Betriebsart 2 werden die Palettencodes 0 bis 15 verwendet und wie folgt in vier Blöcke aufgeteilt.

Palettenblock-Nr.	0	1	2	3
Palettencode	0 bis 3	4 bis 7	8 bis 11	12 bis 15

Die Anfangseinstellung der Palettenblocknummer ist 0 und die Anfangseinstellung der Farbcode-Zuweisungen zu den Palettencodes entspricht der nachstehenden Aufstellung.

Palettencode	Palettenfarbe (Farbe)
0	0 (schwarz)
1	1 (blau)
2	2 (rot)
3	3 (magenta)

Die Palettenblocknummer kann mit dem INIT-Befehl verändert werden.

Beispiel:

INIT "CRT:Bl" ..... Verändert die Palettenblocknummer nach 1.

Die zum gegenwärtigen Palettenblock gehörenden Nummern können als Palettencodes in Befehlen und Anweisungen verwendet werden. Bestimmte Befehle und andere Anweisungen haben einen Parameter zum Spezifizieren eines Palettencodes oder Farbcodes. Die zur gegenwärtigen Palettenblocknummer gehörenden Nummern werden als Palettencodes aufgefaßt und die anderen Nummern als Farbcodes. Zur Veranschaulichung des oben Erklärten soll das folgende Beispiel helfen.

```
10 INIT "CRT:M2,B1" — Betriebsart 2, Palettenblock-Nr. = 1.
20 PAL 4,12 — Weist dem Palettencode 4 den Farbcode 12 zu.
30 PAL 4,10 — Weist dem Palettencode 5 den Farbcode 10 zu.
40 PAL 6,8 — Weist dem Palettencode 6 den Farbcode 8 zu.
50 PAL 7,6 — Weist dem Palettencode 8 den Farbcode 6 zu.
```

Nach Ausführung des obigen Programms zeichnet

```
LINE [5,0] 10,20,100,50
```

eine Linie in hellrot. In diesem Falle wird der erste Parameter als Palettencode verstanden.

```
LINE [5,0] 10,20,100,50
```

Zeichnet die gleiche Linie in blau. In diesem Falle wird der erste Parameter als Farbcode verstanden.

```
INIT "CRT:B0"
```

Wenn der obige Befehl nach Ausführung des obigen Programms ausgeführt wird, so wird ein anderes Ergebnis erhalten.

```
LINE [5,0] 10,20,100,50
```

Zeichnet eine Linie in cyan. In diesem Falle wird der erste Parameter als Farbcode verstanden.

```
LINE [1,0] 10,20,100,50
```

Zeichnet eine Linie in blau. In diesem Falle wird der erste Parameter als Palettencode verstanden. Wenn INIT "CRT:B0" ausgeführt wird, werden die einstellbaren Palettencode wie folgt geändert.

Palettencode		Palettencode
5		0
6	INIT"CRT:B0"	1
7	.....>	2
8		3

(6) Wiederherstellung der Anfangseinstellungen

Die Ausführung einen INIT Anweisung zur Einstellung eines neuen Anzeigenformats stellt die Ausgangseinstellung des Farbcodes ein.
Die Ausgangseinstellungen werden ebenfalls wieder hergestellt, wenn eine INIT Anweisung für Neueinstellung eines Palettenblocks bei der 16-Farben-Betriebsart ausgeführt wird.

(7) Logische Summe der Farben

Einige Grafik-Anweisungen wie COLOR, SET und LINE haben Betriebsarten-Parameter.
Wenn als Betriebsarten-Parameter 0 spezifiziert wird, so wird die sich ergebende Farbe durch den Palettencode-Parameter spezifiziert.
Wenn als Betriebsarten-Parameter 1 spezifiziert ist, so wird die sich ergebende Farbe jedes einzelnen Punktes durch den Palettencode spezifiziert, der der logischen Summe aus dem vorherigen Palettencode für diesen Punkt und dem spezifizierten Palettencode entspricht. Wenn z.B. der gegenwärtige Palettencode für den Punkt an (50,50) 2 ist und der spezifizierte Palettencode ist 1, so ist der sich ergebende Palettencode 3.

0010 (binär für 2) oder 0001 (binär für 1) → 0011 (binär für 3)

Nachstehend ist eine Tabelle der logischen Summen abgebildet.

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	1	1	3	3	5	5	7	7	9	9	11	11	13	13	15	15
2	2	3	2	3	6	7	6	7	10	11	10	11	14	15	14	15
3	3	3	3	3	7	7	7	7	11	11	11	11	15	15	15	15
4	4	5	6	7	4	5	6	7	12	13	14	15	12	13	14	15
5	5	5	7	7	5	5	7	7	13	13	15	15	13	13	15	15
6	6	7	6	7	6	7	6	7	14	15	14	15	14	15	14	15
7	7	7	7	7	7	7	7	7	15	15	15	15	15	15	15	15
8	8	9	10	11	12	13	14	15	8	9	10	11	12	13	14	15
9	9	9	11	11	13	13	15	15	9	9	11	11	13	13	15	15
10	10	11	10	11	14	15	14	15	10	11	10	11	14	15	14	15
11	11	11	11	11	15	15	15	15	11	11	11	11	15	15	15	15
12	12	13	14	15	12	13	14	15	12	13	14	15	12	13	14	15
13	13	13	15	15	13	13	15	15	13	13	15	15	13	13	15	15
14	14	15	14	15	14	15	14	15	14	15	14	15	14	15	14	15
15	15	15	15	15	15	15	15	15	15	15	15	15	15	15	15	15

Das oben Gesagte gilt jedoch nicht für die RESET und BLINE-Anweisung. Für diese Anweisungen wird die sich ergebende Farbe mit dem Farbcode spezifiziert, der durch (Anzahl der Farben, die in der gegenwärtigen Farbanzeige-Betriebsart spezifiziert werden) – 1 – (spezifizierter Palettencode) angegeben wird, wobei der Betriebsarten-Parameter 0 ist. Wenn z.B. als Farbanzeige-Betriebsart die Betriebsart 1 aktiviert ist, in der maximal 4 Farben spezifiziert werden können, führt das Spezifizieren von Farbpalettencode 1, wie nachstehend gezeigt, zu Farbpalettencode 2.

4 – 1 – 1 = 2

Wenn der Betriebsarten-Parameter 1 ist, wird der resultierende Palettencode durch die logische Summe des vorherigen Palettencodes und die (Anzahl der Farben, die in der gegenwärtigen Farbanzeige-Betriebsart spezifiziert werden können) – 1 – (spezifizierter Palettencode) erhalten. Z.B. erhält man durch Spezifizieren von Palettencode 1 mit einem vorherigen Palettencode 2, wie nachstehend dargestellt, den Palettencode 2:

2 ODER 2 → 2

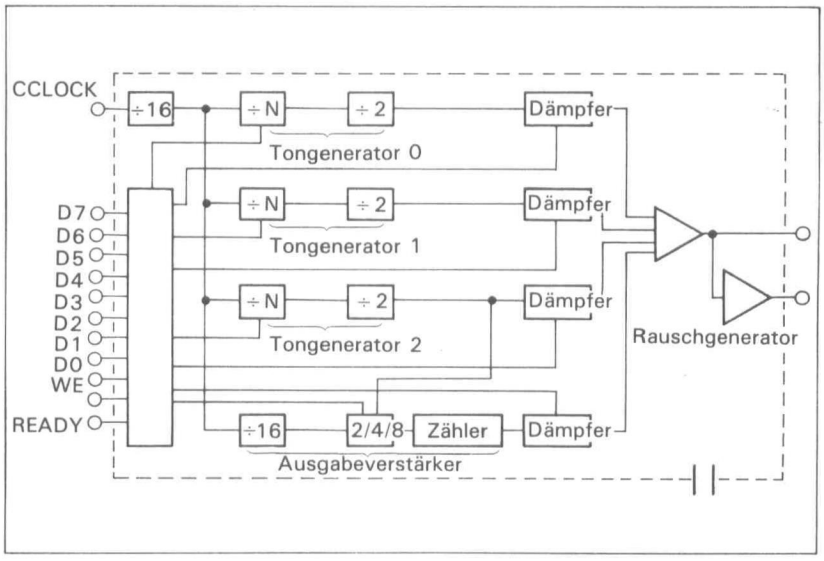
Anhang B Programmierbarer Tongenerator

Der MZ-800 hat einen eingebauten programmierbaren Tongenerator (PSG), der die Erzeugung von 3-Ton-Akkorden über 6 Oktaven ermöglicht. Als PSG wird ein SN76489AN verwendet.
Der PSG kann durch Übertragung von Daten an die I/O-Port-Adresse $F2 gesteuert werden.

(1) Beschreibung des PSG

Der SN76489AN hat 8 interne Register und steuert 3 Tongeneratoren und einen Rauschgenerator.

Blockschaltbild



Das interne Register kann durch Vorgabe der Bits D4 bis D6 im ersten Byte der Ausgabedaten gewählt werden. Die Funktionen der einzelnen Register sind nachstehend aufgeführt.

D6	D5	D4	Funktion
0	0	0	Frequenz von Ton 0
0	0	1	Lautstärke von Ton 0
0	1	0	Frequenz von Ton 1
0	1	1	Lautstärke von Ton 1
1	0	0	Frequenz von Ton 2
1	0	1	Lautstärke von Ton 2
1	1	0	Rauschregelung
1	1	1	Rauschlautstärke

(2) Einstellung der Tonfrequenz

Die Tonfrequenz wird mit den folgenden 2-Byte-Daten eingestellt.

Erstes Byte

D7	D6	D5	D4	D3	D2	D1	D0
1	Registerwahl			Frequenz 1 (niedere 4 Bits)			

Zweites Byte

D7	D6	D5	D4	D3	D2	D1	D0
0	X	Frequenz (höhere 6 Bits)					

Frequenz $= (3.55 \times 10^3)/(32 \times n)$ kHz

n ist eine 10-Bit-Binärzahl, die durch D0 bis D3 des ersten Byte und D0 bis D5 des zweiten Byte dargestellt werden.

(3) Rauscherzeugung

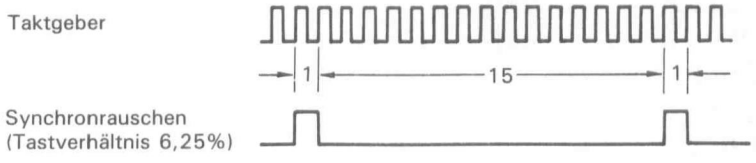
Durch Ausgabe der folgenden 1-Byte-Daten an den I/O-Port kann Synchronrauschen oder weißes Rauschen erzeugt werden.

D7	D6	D5	D4	D3	D2	D1	D0
1	Registerwahl			X	↑ Rauschtyp	Frequenz	

* Der Rauschtyp wird durch D2 spezifiziert.
0: Synchronrauschen
1: Weißes Rauschen

* Die Rauschfrequenz wird durch D0 und D1 vorgegeben. Das Verhältnis zwischen Rauschfrequenz und Einstellung von D0 und Dl ist wie folgt.

Dl	D0	Frequenz
0	0	6.93 $(= 3.55 \times 10^3/512)$ kHz
0	1	3.47 $(= 3.55 \times 10^3/1024)$ kHz
1	0	1.73 $(= 3.55 \times 10^3/2048)$ kHz
1	1	Gleich wie für den durch Tongenerator #3 erzeugten Ton.



* Wenn weißes Rauschen spezifiziert ist, weist die ausgegebene Wellenform willkürliche Muster auf und es wird ein Ton erzeugt, dessen Spektrum fast das gleiche ist wie bei der Erzeugung von weißem Rauschen.

* Wenn die Rausch-Betriebsarten 0,1,1 spezifiziert sind, wird die Frequenz durch die folgende Formel bestimmt und ein Ton mit niedrigem Impuls kann erzeugt werden.

$$f = N/32 \times n \times 16$$

* Wenn man in der Rausch-Betriebsarten-Spezifikation Ton 2 spezifiziert, muß man den Ausgang von Ton 2 ausschalten.

(4) Einstellung der Lautstärke

LS = 1

D7	D6	D5	D4	D3	D2	D1	D0
1	Registerwahl			Dämpfung			

Die Lautstärke wird mit der Dämpfung spezifiziert.

D3	D2	D1	D0	Dämpfung (dB)
0	0	0	0	0
0	0	0	1	2
0	0	1	0	4
0	0	1	1	6
0	1	0	0	8
0	1	0	1	10
0	1	1	0	12
0	1	1	1	14

D3	D2	D1	D1	Dämpfung (dB)
1	0	0	0	16
1	0	0	1	18
1	0	1	0	20
1	0	1	1	22
1	1	0	0	24
1	1	0	1	26
1	1	1	0	28
1	1	1	1	OFF

Anhang C Reservierte Wörter

ABS
AND
ASC
ATN
AUTO
AXIS
BLINE
BOOT
BOX
BYE
CHAIN
CHR$
CIRCLE
CLOSE #
CLR
CLS
COLOR
CONSOLE
CONT
COS
CSRH
CSRV
CURSOR
DATA
DEF
DEFAULT
DELETE
DIM
DIR
ELSE
END
EOF
ERL
ERN
ERROR
EXP
FN
FOR
GET
GOSUB
GOTO
GPRINT
HCOPY
HSET
IF
INIT
INP@
INPUT
INPUT #
INT
KEY
KEYLIST
KILL #
LABEL
LEFT$
LEN
LET
LIMIT
LINE
LIST
LIST/P
LN
LOAD
LOG
MERGE
MID$
MUSIC
NEW
NEXT
NOISE

NOT
OFF
ON
OR
OUT@
PAGE
PAI
PAINT
PAL
PATTERN
PCIRCLE
PCOLOR
PEEK
PHOME
PLINE
PLOT
PMODE
PMOVE
POINT
POKE
POSH
POSITION
POSV
PRINT
PRINT #
PSKIP
PTEST
RAD
READ
REM
RENAME
RENUM
RESET
RESTORE
RESUME
RETURN
RIGHT$
RLINE
RMOVE
RND
ROPEN #
RUN
SAVE
SEARCH
SEARCH/P
SET
SGN
SIN
SIZE
SOUND
SPC
SQR
STICK
STOP
STR$
STRIG
SYMBOL
TAB
TAN
TEMPO
THEN
TI$
TROFF
TRON
USING
USR
VAL
VERIFY
WAIT
WOPEN #
XOR

Anhang D Konsolensteuercodes

Einige der Codes von 0 bis 26 werden, wie nachstehend gezeigt, zur Steuerung des Computerbetriebs verwendet. Diese Steuercodes können direkt über die Tastatur oder unter Verwendung der PRINT-Anweisung (in der Code-Spalte durch ein Sternsymbol (*) gezeigt) in den Computer eingegeben werden.

Steuercode-Tabelle

Code (dez.)	Tasteneingabe	Funktion
5	CTRL + E	Zur Eingabe von Kleinbuchstaben mit den Zeichentastesn.
6	CTRL + F	Zur Eingabe von Großbuchstaben mit den Zeichentasten.
13 (*)	CTRL + M	Wagenrücklauf. CR
16 (*)	CTRL + P	Löscht das Zeichen in Cursorposition. DEL
17 (*)	CTRL + Q	Abwärtsbewegung des Cursors. ↓
18 (*)	CTRL + R	Aufwärtsbewegung des Cursors. ↑
19 (*)	CTRL + S	Bewegt den Cursor nach rechts. →
20 (*)	CTRL + T	Bewegt den Cursor nach links. ←
21 (*)	CTRL + U	Bewegt den Cursor in Ausgangsposition. HOME
22 (*)	CTRL + V	Löscht den Bildschirm. CLR
23 (*)	CTRL + W	Aktiviert die Grafik-Betriebsart der Tastatur. GRAPH
24 (*)	CTRL + X	Einfügen einer Leerstelle in Cursorposition. INST
25 (*)	CTRL + Y	Aktiviert die Normal-Betriebsart der Tastatur. ALPHA

Anhang E Einschränkungen beim Gebrauch von Datei-I/O-Befehlen und Anweisungen.

Datei-I/O-Befehle und -Anweisungen.
Datei-I/O-Befehle und Anweisungen können nicht für alle Speichergeräte verwendet werden. Die nachstehende Tabelle zeigt Einschränkungen bei ihrer Verwendung.

Gerät / Befehl/ Anweisung	CTM: (Datenrecorder)	RAM: (RAM Datet-Steckkarte)	CRT: (Anzeige)	LPT: (Drucker)	RS1: bis RS2: (RS-232C)
INIT	○	×	×	×	×
DEFAULT	×	×	×	×	×
DIR	○	×	○	○	○
RUN	×	×	○	○	○
LOAD	×	×	○	○	○
SAVE	×	×	○	○	○
DELETE	○	×	○	○	○
RENAME	○	×	○	○	○
CHAIN	×	×	○	○	○
MERGE	×	×	○	○	○
WOPEN #	×	×	○	○	×
PRINT #	×	×	○	○	×
ROPEN #	×	×	○	○	×
INPUT #	×	×	○	○	×
CLOSE #	×	×	○	○	×
KILL #	×	×	○	○	×

×: Kann verwendet werden.
○: Kann nicht verwendet werden.

Darüberhinaus gilt für CMT: und RS1: bis RS2, daß nur eine Datei gleichzeitig geöffnet werden kann.

Anhang F ROM-Monitor-Unterprogramme

Die folgenden Unterprogramme sind für den Monitor 9Z-504M vorgesehen. Jeder Unterprogrammname steht symbolisch für die Funktion des entsprechenden Unterprogramms. Diese Unterprogramme können von Anwenderprogrammen her aufgerufen werden. Die gespeicherten Register sind diejenigen, deren Inhalt bei Rückkehr der Ablaufsteuerung zum aufrufenden Programm wiederhergestellt werden. Der Inhalt der anderen Register wird durch Ausführung des Unterprogramms verändert.

Name und Einsprungstelle (hex.)	Funktion	Gespeichertes Register
CALL LETNL (0006)	Bewegt den Cursor zum Anfang der nächsten Zeile.	Alle außer AF
CALL PRNT (000C)	Zeigt in Cursorposition eine Leerstelle an.	Alle außer AF
CALL PRNT (0012)	Anzeige des Zeichens, das dem im ACC gespeicherten Wert im ASCII-Codedarstellung in Cursorposition entspricht. ASCII-Code siehe Anhang J. Wenn Code 0D (Wagenrücklauf) oder 11 bis 16 (Cursorsteuercodes) eingegeben wird, erfolgt keine Zeichenanzeige, stattdessen wird die entsprechende Funktion ausgeführt (Wagenrücklauf bei OD und Cursorbewegung bei 11 bis 16)	Alle außer AF
CALL MSG (0015)	Anzeige einer Meldung ab Cursorposition. Die Anfangsadresse des Bereiches, in dem die Meldung gespeichert ist, muß vor Aufrufen dieses Unterprogramms im DE-Register vorgegeben sein und die Meldung muß mit einem Wagenrücklauf-Code (OD) enden. Wagenrücklauf wird nicht ausgeführt. Wenn Cursorsteuercodes (11 bis 16) in der Meldung enthalten sind, wird der Cursor bewegt.	Alle Register
CALL BELL (003E)	Der Ton A (ca. 880 Hz.) ertönt kurz	Alle außer AF
CALL MELDY (0030)	Gemäß den Musikdaten, die im Speicherbereich ab der durch das DE-Register angezeigten Adresse gespeichert sind, erfolgt Musikwiedergabe. Die Musikdaten müssen im gleichen Format vorliegen wie für die MUSIC-Anweisung von BASIC, und müssen mit 0D oder C8 enden. Wenn der Spielbetrieb beendet ist, erfolgt Rückkehr der Ablaufsteuerung an das aufrufende Programm, wobei die C-Flag auf 0 gesetzt ist; wenn die Wiedergabe durch Drücken der [BREAK]-Taste unterbrochen wird, erfolgt Rückkehr der Ablaufsteuerung, wobei die C-Flag auf 1 gesetzt ist.	Alle außer AF
CALL XTEMP (0041)	Das musikalische Tempo wird gem. den im Akumulator (ACC) gespeicherten Tempodaten eingestellt. ACC ← 01 Langsamstes Tempo ACC ← 04 Mittleres Tempo ACC ← 07 Schnellstes Tempo Bitte beachten, daß die Daten im Akumulator nicht dem ASCII-Code entsprechend 1 bis 7 sind, sondern Binärcode.	Alle Register

Name und Einsprungstelle (hex.)	Funktion	Gespeichertes Register
CALL MSTA (0044)	Generiert einen kontinuierlichen Ton der spezifizierten Frequenz. Die Frequenz wird durch die folgende Gleichung gegeben: frequ. = 895 kHz/nn' Hierbei ist nn' eine 2-Byte-Zahl, die in den Adressen 11A1 und 11A2 (n in 11A2 und n' in 11A1) gespeichert wird.	BC und DE
CALL MSTP (0047)	Stoppt den durch das CALL MSTA Unterprogramm erzeugten Ton.	Alle außer AF
CALL TIMST (0033)	Zum Einstellen und Starten der eingebauten Uhr. Vor Aufrufen dieses Unterprogramms müssen die Register wie folgt eingestellt werden. ACC ← 0 (AM), ACC ← 1 (PM) DE ← 4-stellige Hexadezimalzahl (die die Zeit in Sekundenform ausdrückt).	Alle außer AF
CALL TIMRD (003B)	Ablesen der eingebauten Uhr und Angabe der Zeit wie folgt. ACC ← 0 (AM), ACC ← 1 (PM) DE ← 4-stellige Hexadezimalzahl zur Darstellung der Zeit in Sekunden.	Alle außer AF und D.
CALL BRKEY (001E)	Überprüft, ob die [SHIFT] und [BREAK]-Tasten beide gedrückt sind. Wenn beide Tasten gleichzeitig drückt sind, wird die die Z-Flag gesetzt, anderenfalls wird sie rückgesetzt.	Alle außer AF
CALL GETL (0003)	Liest eine Zeile Daten von der Tastatur und speichert sie im Speicherbereich ab der im DE-Register angezeigten Adresse. Durch Drücken der [CR]-Taste stoppt dieses Unterprogramm das Lesen von Daten und fügt einen Wagenrücklaufcode (0D) am Ende der gelesenen Daten an. Max. 80 Zeichen (einschl. Wagenrücklaufcode) können in einer Zeile eingegeben werden. Eingetastete Zeichen erscheinen auf der Anzeige (Echo) und die Zeile kann Cursorsteuercodes enthalten. Durch gleichzeitiges Drücken der [SHIFT]- und [BREAK]-Taste wird der BREAK-Code in der im DE-Register stehenden Adresse und ein Wagenrücklaufcode in der nachfolgenden Adresse gespeichert.	Alle Register
CALL GETKY (001B)	Liest einen Zeichencode von der Tastatur. Wenn keine Taste gedrückt wird, erfolgt Rückkehr der Ablaufsteuerung an das aufrufende Programm, wobei als Inhalt des ACC 00 vorgegeben wird. Es werden keine speziellen Vorkehrungen getroffen, um Datenlesefehler aufgrund von Tastenprellen zu vermeiden und die eingegebenen Zeichen erscheinen nicht zur Bestätigung auf dem Bildschirm. Bei Betätigung einer der Spezialtasten (z.B. [DEL] oder [CR]) erfolgt durch dieses Unterprogramm die Rückgabe eines Codes, der sich, wie nachstehend gezeigt, vom ASCII-Code unterscheidet, an den Akkumulator. Hier werden Anzeigecodes verwendet, um die im Zeichengenerator gespeicherten Zeichen zu adressieren, sie unterscheiden sich von den ASCII-Codes.	Alle außer AF

Name und Einsprungstelle (hex.)	Funktion			Gespeichertes Register
Mit GETKY gelesene Spezialtaste	Spezialtaste	Im ACC vorgegebener Code	Anzeigecode	
	DEL	60	C7	
	INST	61	C8	
	ALPHA	62	C9	
	BREAK	64	CB	
	CR	66	CD	
	↓	11	C1	
	↑	12	C2	
	→	13	C3	
	←	14	C4	
	HOME	15	C5	
	CLR	16	C6	
CALL ASC (03DA)	Bestimmt das ASCII-Zeichen gemäß der durch die 4 niederen Bits der Daten im Akkumulator vorhandenen Hexadezimalzahl.			Alle außer AF
CALL HEX (03F9)	Wandelt die im Akkumulator gespeicherten 8 Datenbits in eine Hexadezimalzahl um (vorausgesetzt, daß es sich bei den Daten um ein ASCII-Zeichen handelt) und setzt diese Hexadezimalzahl in den 4 niederen Bits des Akkumulators. Wenn eine Hexadezimalzahl im Akkumulator vorliegt, wird die C-Flag auf 0 gesetzt, anderenfalls wird sie auf 1 gesetzt.			Alle außer AF
CALL HLHEX (0410)	Wandelt eine Zeichenfolge von 4 ASCII-Zeichen in eine Hexadezimalzahl um und setzt sie im HL-Register. Die Aufruf- und Rücksprungbedingungen sind wie folgt. DE ← Anfangsadresse des Speicherbereiches, der die ASCII-Zeichenfolge enthält. (d.h.,"3" "1" "A" "5") CALL HLHEX ↑—DE CF = 0 HL ← Hexadezimalzahl (d.h HL = 31A5H) CF = 1 Der Inhalt von HL ist nicht gewährleistet.			Alle außer AF und HL
CALL 2HEX (041F)	Wandelt eine Folge von 2 ASCII-Zeichen in eine Hexadezimalzahl um und setzt sie in den Akkumulator. Die Aufruf- und Rücksprungbedingungen sind wie folgt. DE ← Anfangsadresse des Speicherbereiches, der die ASCII-Zeichenfolge enthält (z.B. "3" "A") CALL 2HEX ↑—DE CF = 0 ACC-Hexadezimalzahl (d.h. ACC = 3H) CF = 1 Der Inhalt des ACC ist nicht gewährleistet			Alle außer AF und DE
CALL ??KEY (09B3)	Bewirkt Blinken des Cursors als Aufforderung zur Tasteneingabe. Durch Drücken einer Taste wird der entsprechende Anzeigecode in den Akkumulator gesetzt und die Ablaufsteuerung an das aufrufende Programm zurückgegeben.			Alle außer AF
CALL ?ADCN (0BB9)	Zur Umwandlung von ASCII-Codes in Anzeigecodes. Die Aufruf- und Rücksprungbedingungen sind wie folgt. ACC ← ASCII-Code CALL ?ADCN ACC ← Anzeigecode			Alle außer AF

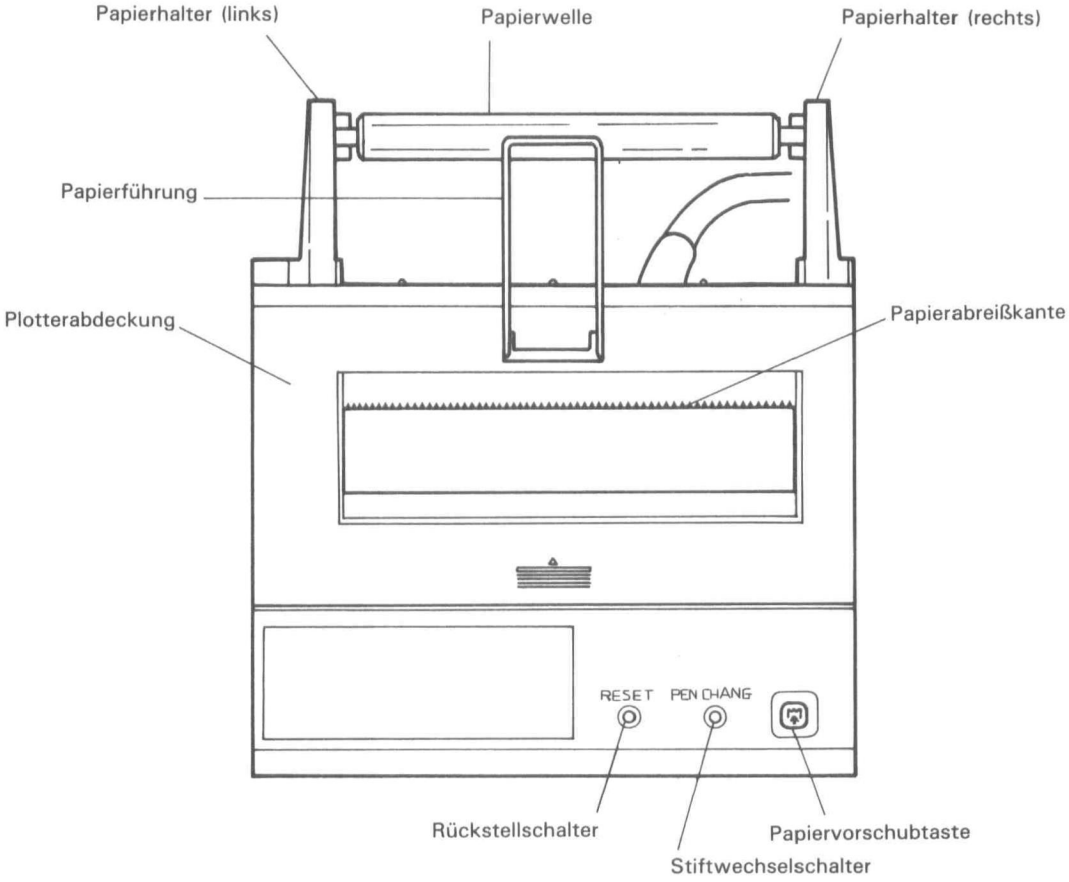
<table>
<tr><th>Name und Einsprungstelle (hex.)</th><th>Funktion</th><th>Gespeichertes Register</th></tr>
<tr><td>CALL ?DACN
(0BCE)</td><td>Wandelt Anzeigecodes in ASCII-Codes um. Die Aufruf- und Rücksprungbedingungen sind wie folgt.
ACC ←Anzeigecode
CALL ?DACN
ACC ←ASCII-Code</td><td>Alle außer AF</td></tr>
<tr><td>CALL ?BLNK
(0DA6)</td><td>Erkennt die Bildaustastperiode. Am Anfang der Bildaustastperiode erfolgt Rückgabe der Ablaufsteuerung an das aufrufende Programm.</td><td>Alle Register</td></tr>
<tr><td>CALL ?DPCT
(0DDC)</td><td>Die Anzeige wird wie folgt gesteuert.
<table>
<tr><th>ACC</th><th>Steuerung</th><th>ACC</th><th>Steuerung</th></tr>
<tr><td>$C0_H$</td><td>Scrolling</td><td>$C6_H$</td><td>Gleich wie die [CLR]-Taste.</td></tr>
<tr><td>$C1_H$</td><td>Gleich wie die [↓] Taste.</td><td>$C7_H$</td><td>Gleich wie die [DEL]-Taste.</td></tr>
<tr><td>$C2_H$</td><td>Gleich wie die [↑] Taste.</td><td>$C8_H$</td><td>Gleich wie die [INST]-Taste.</td></tr>
<tr><td>$C3_H$</td><td>Gleich wie die [→] Taste.</td><td>$C9_H$</td><td>Gleich wie die [ALPHA]-Taste.</td></tr>
<tr><td>$C4_H$</td><td>Gleich wie die [←] Taste.</td><td>CD_H</td><td>Gleich wie die [CR]-Taste.</td></tr>
<tr><td>$C5_H$</td><td>Gleich wie die [HOME]-Taste.</td><td></td><td></td></tr>
</table></td><td>Alle Register</td></tr>
<tr><td>CALL ?PONT
(0FB1)</td><td>Setzt die gegenwärtige Cursorposition im HL-Register. Die Rücksprungbedingungen sind wie folgt.
CALL? PONT
HL (Cursorposition) (binär)</td><td>Andere als AF und HL</td></tr>
</table>

Anhang G Herstellung einer Sicherungskopie des BASIC-Interpreters

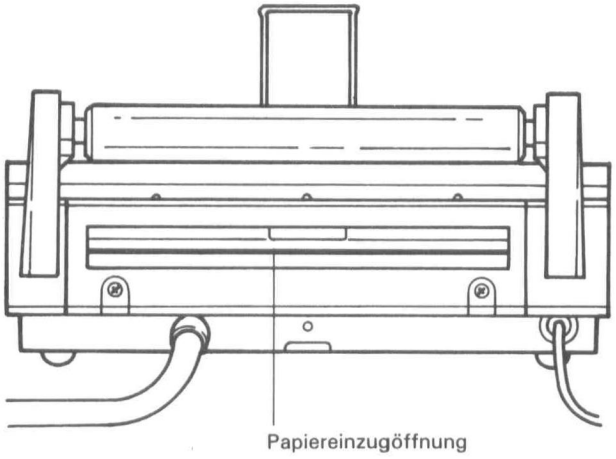
Es ist möglich, daß die Cassette, die den BASIC-Interpreter enthält, versehentlich beschädigt wird. In diesem Falle kann man den Computer nicht verwenden. Um dies zu vermeiden, empfiehlt es sich, eine Kopie des BASIC-Interpreters herzustellen. Nach Herstellung der Sicherungskopie verwahrt man die Originalcassette an einen sicheren Ort und verwendet stattdessen im täglichen Einsatz die Sicherungskopie. Nachstehend ist die Verfahrensweise zur Herstellung einer Sicherungskopie beschrieben.

1) Eine neue Cassette vorbereiten.
2) Den MZ-800 einschalten und zum Starten des Monitors die [M]-Taste drücken.
3) Die den BASIC-Interpreter enthaltende Cassette in den Datenrecorder einlegen und den folgenden Monitor-Befehl eingeben.
   *GE807 [CR]
4) Wenn "↓PLAY" angezeigt wird, drückt man die Wiedergabetaste (PLAY) des Datenrecorders, um den BASIC-Interpreter in den Speicher einzulesen.
5) Wenn das Bereitschaftszeichen (*) erscheint, tauscht man die Cassette gegen eine neue aus und gibt den folgenden Monitor-Befehl ein.
   *GE80A [CR]
6) Wenn "↓RECORD.PLAY" angezeigt wird, drückt man die Aufnahmetaste (RECORD) des Datenrecorders, um den BASIC-Interpreter auf die neue Cassette zu schreiben.
7) Wenn das Bereitschaftszeichen (*) erscheint, das Band zurückspulen. Daraufhin den folgenden Monitor-Befehl eingeben:
   *GE80D [CR]
8) Wenn "↓PLAY" angezeigt wird, drückt man die Wiedergabetaste (PLAY) um zu verifizieren, daß die Bandaufzeichnung mit dem Speicherinhalt übereinstimmt.
9) Wenn die Meldung "CHECK SUM ER." erscheint, ist Wiederholung der Schritte 3 bis 8 erforderlich. Wenn die Meldung "OK." erscheint, ist der Kopiervorgang abgeschlossen.

Anhang H Farbplotter



(Ansicht von oben)



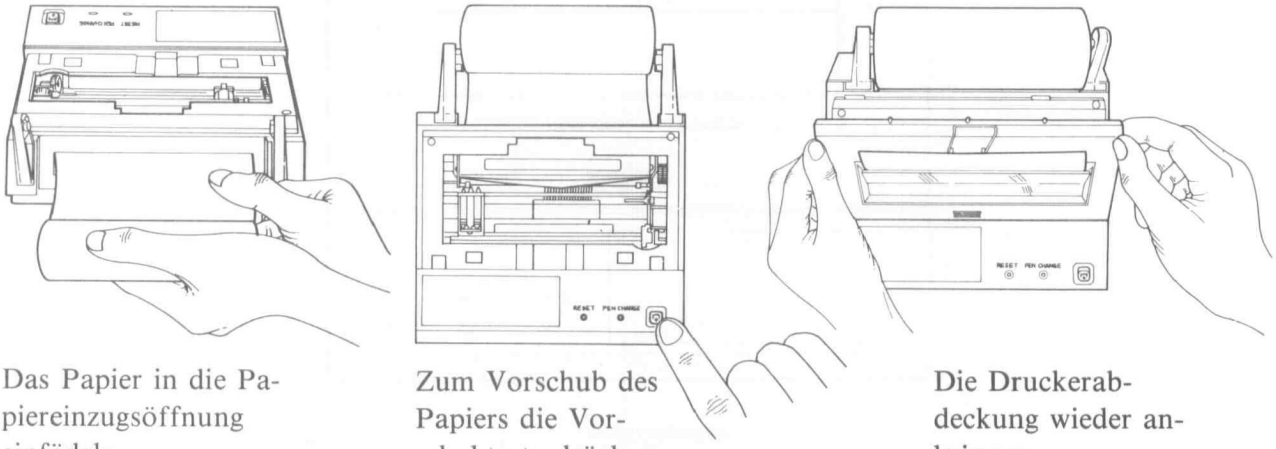
(Ansicht von hinten)

Hinweis:
Für den Schutz der Druckermechanismus wurde ein Schutzblech eingefügt. Das Blech entfernen, indem vor Verwendung die papiervorschubtaste ( ) gedrückt wird.

■ Einlegen der Papierrolle

1. Die Plotterabdeckung entfernen
2. Das Ende der Papierrolle in einer geraden Linie abschneiden und den Papieranfang in die Papiereinzugöffnung einschieben. (Dabei darauf achten, daß der Papieranfang nicht gefaltet oder zerknittert wird).
3. Den Netzschalter des MZ-800 einschalten und die -Taste (Papiervorschubtaste) drücken, so daß das Papier so weit vorgeschoben wird, bis es 3 bis 5 cm über die Auslaßöffnung herausragt.
4. Die Papierwelle in die Rolle einführen und an den Papierhaltern anbringen.
5. Die Plotterabdeckung so anbringen, daß das Papieranfangsstück durch die Abschneidevorrichtung läuft.

- Um die Rolle für den Austausch aus dem Plotter zu entnehmen, schneidet man es an der Einlaßseite in einer geraden Linie ab und drückt die Papiervorschubtaste ( ).



Das Papier in die Papiereinzugsöffnung einfädeln.

Zum Vorschub des Papiers die Vorschubtaste drücken.

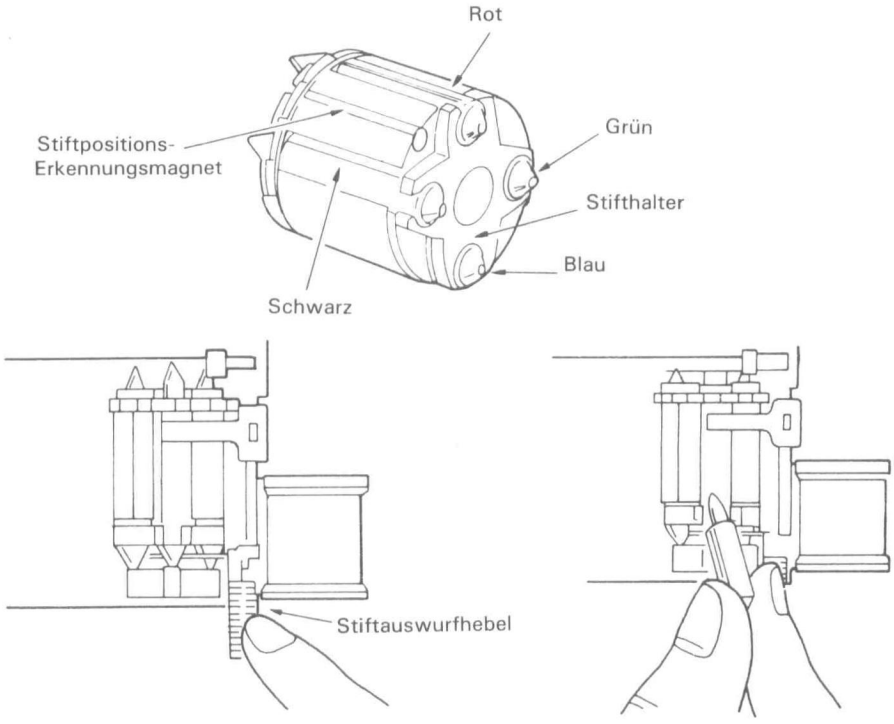
Die Druckerabdeckung wieder anbringen.

■ Papierrollen für Computer der Serie MZ-800 sind beim SHARP-Fachhändler erhältlich. Nur das vorgeschriebene Papier verwenden.

Die Papierlänge beträgt 23 bis 25 Meter und der max. Durchmesser der Rolle, die eingelegt werden kann, beträgt ϕ 50 mm. Bei Verwendung von Papier mit einem größeren Rollendurchmesser wird die Druckqualität beeinträchtigt.

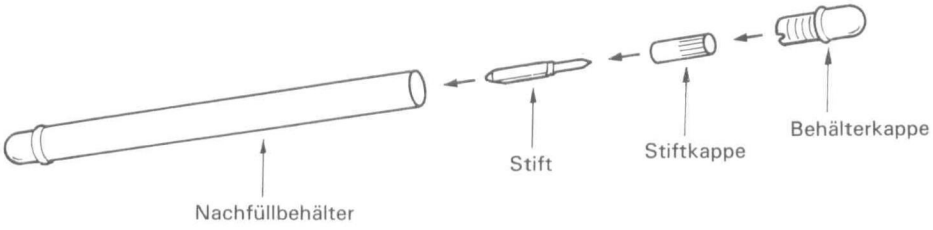
■ Einsetzen/Austauschen der Stifte

1. Die Plotterabdeckung entfernen und den PEN CHANGE-Schalter mit einem Kugelschreiber o. dgl. drücken, dadurch wird die Stifthaltertrommel nach rechts bewegt und die Stifte können ausgetauscht werden.
2. Zum Auswerfen des z.Z. oben befindlichen Stifts in der Stifthaltertrommel betätigt man den Stiftauswurfhebel. Während man den Auswurfhebel drückt, empfiehlt es sich, einen Finger mit leichtem Druck auf dem Stift ruhen zu lassen, um zu verhindern, daß der Stift in den Plotter fällt.
3. Einen neuen Stift einsetzen.
4. Um einen weiteren Stift in die obere Position der Stifthaltertrommel zu bringen, den Stiftwechselschalter (PEN CHANGE) nochmals drücken.
5. Auf diese Weise kann man alle vier Stifte (schwarz,blau, grün und rot) in der gleichen Weise austauschen. Wenn dieser Vorgang abgeschlossen ist, drückt man den RESET-Schalter, um den Plotter zum Drucken mit dem schwarzen Stift vorzubereiten.
   Zur Bestätigung, daß alle Farben korrekt ausgedruckt werden, den BASIC-PTEST-Befehl ausführen.



■ Aufbewahrung der Stifte

Die Stifte nur einsetzen, wenn der Drucker verwendet wird.
Wenn der Drucker nicht verwendet wird, entnimmt man die Stifte und setzt die Kappen auf, dann bewahrt man sie im Nachfüllbehälter auf, anderenfalls trocknen sie aus.



Hinweis:
Da die Kugelminenstifte wasserlösliche Tinte enthalten, und die Tinte verschmiert werden kann, wenn das Papier naß wird, sollte man das Papier mit gebührender Vorsicht behandeln.

■ Für den Austausch der Kugelminenstifte wendet man sich an den Fachhändler, bei dem man den Plotter gekauft hat.

- EA-850B (schwarz: 4 Stifte)
- EA-850C (schwarz, blau, grün und rot; 4 Stifte, einer von jeder Farbe)

■ Selbsttest

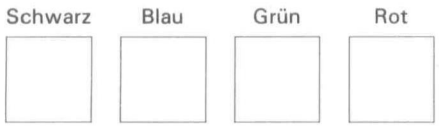
Der Drucker hat eine Selbsttestfunktion. Die [⇧] Taste (Papiervorschub-Taste) gedrückt halten und das Gerät einschalten, woraufhin der Selbsttest beginnt. Es wird empfohlen, beim Starten des Gerätes den Selbsttest auszuführen. Desgleichen sollte man den Selbsttest nach Austausch der Stifte ausführen.

Hinweis:
Bei Ausführung des Selbsttests auf jeden Fall das Schnittstellenkabel abtrennen.

Anhang I Farbplotter-Steuercodes

1 In der Textbetriebsart verwendete Steuercodes

- Textcode ($01)
  Aktiviert die Text-Betriebsart.
- Grafikcode ($02) ..... Gleich wie die BASIC PMODE GR-Anweisung
  Aktiviert die Grafik-Betriebsart.
- Zeile aufwärts ($03) ..... Gleich wie die BASIC PSKIP-Anweisung.
  Bewegt das Papier um eine Zeile in Gegenrichtung. Abnahme des Zeilenzählers um eins.
- Stifttest ($04) ..... Gleich wie die BASIC PTEST-Anweisung.
  Schreibt die folgenden Muster, dient dazu, den Tintenfluß der Stifte zu starten, gibt daraufhin Zeichengröße = 1 (40 Zeichen/Zeile) und Farbe = 0 vor.



- Skalenverkleinerung ($09) + ($09) + ($09)
  Verkleinert die Zeichengröße von 1 nach 0 (80 Zeichen/Zeile)
- Aufhebung der Verkleinerung ($09) + ($09) + ($0B)
  Vergrößert den Zeichengröße von 0 nach 1 (40 Zeichen/Zeile)
- Zeilenzähler-Einstellung ($09) + ($09) + $(ASCII)_2$ + $(ASCII)_1$ + $(ASCII)_0$ + ($0D)
  ..... Gleich wie die BASIC PAGE-Anweisung.
  Spezifiziert die Anzahl der Zeilen pro Seite, wie durch die 3 Bytes des ASCII-Codes angezeigt. Die max. Anzahl Zeilen pro Seite ist 255. Bei Einschalten des Gerätes oder Rücksetzen des Systems beträgt die Standardvorgabe 66.
- Zeilenvorschub ($0A) ..... Gleich wie die BASIC PSKIP-Anweisung.
  Das Papier wird um eine Zeile in Vorwärtsrichtung bewegt. Der Zeilenzähler wird um 1 erhöht.
- Skalenvergrößerung ($0B)
  Die Zeichengröße wird von 1 nach 2 vergrößert (26 Zeilen/Zeile)
- Aufhebung der Skalenvergrößerung ($0C)
  Die Zeichengröße wird von 2 nach 1 reduziert.
- Wagenrücklauf ($0D)
  Der Wagen wird zur linken Seite des Druckbereiches bewegt.
- Rückschritt ($0E)
  Der Wagen wird um eine Stelle nach links bewegt. Dieser Code wird ignoriert, wenn der Wagen sich an der linken Seite des Druckbereiches befindet.
- Formularvorschub ($0F)
  Bewegt das Papier zum Anfang der nächsten Seite und stellt den Zeilenzähler auf 0 zurück.
- Nächste Farbe ($1D)
  Bewirkt Stiftwechsel auf die nächste Farbe.

2 Zeichengröße

- Bei Einschalten des Gerätes wird als Standardvorgabe automatisch die Zeichengröße 1 (40 Zeichen/Zeile) vorgegeben. Danach kann man sie durch Steuercodes und Steuerbefehle verändern.
- In der Grafik-Betriebsart kann die Skalengröße im Bereich von 0 bis 63 verändert werden.
- Bei Umschaltung der Betriebsart von Grafik nach Text wird als Skalengröße 1 vorgegeben.

3 Befehle der Grafik-Betriebsart

Befehlstyp

Befehlsname	Format	Funktion
LINE TYPE	Lp (p = 0 bis 15)	Spezifiziert den Linientyp (durchgehend oder punktiert) und den Punktabstand. p = 0 : durchgehende Linie, p = 1 to 15 : punktierte Linie
ALL INITIALIZE	A	Aktiviert am Drucker die Text-Betriebsart.
HOME (PHONE)	H	Zum Abheben des Stifts und Rückführung zum Ursprung (Ausgangsposition).
INITIALIZE (HSET)	I	Setzt die gegenwärtige Stiftposition als Ursprung (x = 0, y = 0).
DRAW (LINE)	Dx, y, ... xn, yn ($-999 \leqq x, y \leqq 999$)	Zeichnet Linien von der gegenwärtigen Stiftposition zu den Koordiaten (x_1, y_2), daraufhin zu den Koordinaten (x_2,y_2) usw.
RELATIVE DRAW (RLINE)	JΔx, Δy, ... Δxn, Δyn ($-999 \leqq \Delta x, \Delta y \leqq 999$)	Zeichnet Linien von der gegenwärtigen Stiftposition zu den relativen Koordinaten (Δx_1, Δy_1), dann zu den relativen Koordinaten (Δx_2, Δy_2) usw.
MOVE (MOVE)	Mx, y ($-999 \leqq \Delta x, \Delta y \leqq 999$)	Hebt den Stift ab und bewegt ihn zu den Koordinaten (x,y)
RELATIVE MOVE (RMOVE)	RΔx, Δy ($-999 \leqq \Delta x, \Delta y \leqq 999$)	Hebt den Stift ab und bewegt ihn zu den relativen Koordinaten (Δx, Δy).
COLOR CHANGE (PCOLOR)	Cn (n = 0 bis 3)	Wechselt die Stiftfarbe nach n.
SCALE SET	Sn (n = 0 bis 63)	Spezifiziert die Zeichengröße.
ALPHA ROTATE	Qn (n = 0 bis 3)	Spezifiziert die Richtung, in der die Zeichen gedruckt werden.
PRINT	P$c_1c_2c_3$ ... cn (n = ∞)	Druckt Zeichen.
AXIS (AXIS)	Xp, q, r (p = 0 oder 1) (q = -999 bis 999) (r = 1 bis 255)	Zeichnet eine X-Achse, wenn p = 1 und eine Y-Achse, wenn p = 0. q spezifiziert die Skaleneinteilung und r spezifiziert die Anzahl der zu zeichnenden Skalenmarkierungen.

Befehlsformat

Wie nachstehend gezeigt, gibt es 5 Typen von Befehlsformat.

1. Nur Befehlszeichen (ohne Parameter)
   A, H, I
2. Befehlszeichen plus einen Parameter
   L, C, S, Q
3. Befehlszeichen plus Parameterpaare
   D, J, M, R
   " , " wird zur Abtrennung der Parameter verwendet und mit dem CR-Code wird die Parameterliste beendet.
4. Befehl plus Zeichenfolge
   P
   Die Zeichenfolge wird mit einem CR-Code abgeschlossen.
5. Befehl plus drei Parameter
   X
   " , " wird zum Abtrennen der Parameter verwendet.

Parameter-Spezifikation

1. Führende Leerstellen werden ignoriert.
2. Jede Zahl, der ein " – " vorgestellt ist, wird als negative Zahl behandelt.
3. Wenn die Anzahl der Stellen einer Zahl 3 überschreitet, gelten nur die untersten 3 Stellen.
4. Jeder Parameter wird durch " , " oder einen CR-Code beendet. Wenn in einem Parameter andere Zeichen als Zahlen enthalten sind, werden alle nachfolgenden Zeichen ignoriert bis ein Komma oder ein CR-Code auftritt.
   Beispiel: D␣␣– 135, 21, ...
   Ignoriert.

Wird ignoriert Abgekürzte Formate:

1. Allen Ein-Zeichen-Befehlen kann ohne Eingabe eines CR-Codes ein weiterer Befehl nachgestellt werden.
   Beispiel: "HD100, 200" CR ist gültig und hat die gleiche Funktion wie "H" CR "D100, 200" CR.
2. Befehlen mit einem Parameter können weitere Befehle nachgestellt werden, indem man diese einfach durch ein Komma (",") voneinander trennt.
   Beispiel: L0, Sl, Q0, Cl, D100, 200 CR ist gültig.
3. Ein Befehl mit Parameterpaaren muß mit einem CR-Code abgeschlossen werden.

Datenveränderung aufgrund von Betriebsarten-Umschaltung

Die folgenden Datenveränderungen treten auf, wenn der Drucker von der Grafik-Betriebsart auf die Text-Betriebsart umgeschaltet wird.

- X- und Y-Koordinaten: Y wird auf 0 und der Ursprung wird an den linken Rand des druckbaren Bereiches gesetzt.
- Zeichendruckrichtung: Q wird auf 0 gesetzt.
- Zeichengröße: Die Zeichengröße wird auf 1 eingestellt.
- Die Zeilentyp-Einstellung ist nicht betroffen.

Anhang J Code-Tabellen

■ ASCII-Codetabelle

MSD ist eine Abkürzung für höchstwertige Ziffer und steht für die höheren 4 Bits jedes Codes; LSD ist eine Abkürzung für niedrigstwertige Ziffer und steht für niederen 4 Bits jedes Codes. Code 11H bis 16H sind Cursorsteuercodes. Z.B. wird durch Ausführung von CALL PRNT (ein Monitor-Unterprogramm), wobei 15H im Akkumulator gesetzt ist, der Cursor in die Ausgangsposition zurückgeführt (" H " wird nicht angezeigt).

LSD \ MSD		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
		0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
0	0000			SP	0	@	P					q	n				
1	0001		↓	!	1	A	Q					a					
2	0010		↑	"	2	B	R				e	z	Ü				
3	0011		→	#	3	C	S				`	w	m				
4	0100		←	$	4	D	T				~	s					
5	0101		H	%	5	E	U					u					
6	0110		C	&	6	F	V				t	i		→			
7	0111			'	7	G	W				g		o				O
8	1000			(	8	H	X				h	Ö	l				♣
9	0001			)	9	I	Y					k	Ä				
A	1010			*	:	J	Z				b	f	ö				◆
B	1011			+	;	K	[		°	^	x	v	ä				£
C	1100			,	<	L	\				d						↓
D	1101	CR		—	=	M	]				r	ü	y				
E	1110			.	>	N	↑				p	ß	{				
F	1111			/	?	O	←				c	j					π

■ ASCII-Codetabelle für Farbplotter

Außer den obigen Zeichen können keine sonstigen Grafik-Zeichen gedruckt werden, hingegen wird der entsprechende Hexadezimalcode in einer unterschiedlichen Stiftfarbe ausgedruckt.

MSD / LSD	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0			SP	0	@	P			}		q	n				
1		↓	!	1	A	Q					a					
2		↑	"	2	B	R				e	Z	Ü				
3		→	#	3	C	S				`	w	m				
4		←	$	4	D	T				~	s					
5		H	%	5	E	U					u					
6		C	&	6	F	V				t	¦		→			
7			´	7	G	W				g		O		—		
8			(	8	H	X				h	Ö	!				
9			)	9	I	Y					k	Ä				
A			*	:	J	Z				b	f	Ö				
B			+	;	K	[		▫	^	x	v	ä				£
C			,	<	L	\				d						↓
D			—	=	M	]				r	ü	y				
E			.	>	N	↑				p	ß	{				
F			/	?	O	←				c	j		_			π

Anhang K Fehlermeldungen des Monitors

[Cassettenband]

CMT: Loading error
Auftreten eines Fehlers während des Daten-Ladebetriebs.

CHECK SUM ER.
Auftreten eines Fehlers in der Prüfsumme der geladenen Daten.

Make ready CMT:
Versuchter Zugriff auf Daten ohne Betätigung der [PLAY]-Taste bzw. ohne daß sich eine Cassette im Cassettenlaufwerk befindet.

[MZ-Diskette]

QD: Loading error
Ein Fehler, der während des Ladens von Daten auftrat.

QD: File mode error
Die in der Einschaltsequenz vorgegebene Startdatei war keine OBJ-Datei.

QD: File not found
Die spezifizierte OBJ-Datei wurde nicht gefunden.

QD: Hard err
Ein Hardware-Fehler ist aufgetreten.

Already exist err
Der für den S-Befehl spezifizierte Dateiname ist bereits auf der Diskette vorhanden.

QD: Write protect
Versuchter Zugriff auf eine schreibgeschützte Diskette.

QD: Not ready
Versuchter Zugriff auf eine Diskette bei geöffnetem Laufwerkhebel oder während keine Diskette eingelegt ist.

QD: No file space err
Unzureichende Speicherkapazität beim Versuch, eine Datei mit dem S-Befehl zu speichern.

QD: Unformat err
Die Diskette, auf die der Zugriff erfolgen sollte, ist nicht formatiert.

QD: Bad disk err
Die Diskette, auf die der Zugriff erfolgen sollte, ist defekt.

Make ready QD
Ein Versuch wurde unternommen, auf Daten zuzugreifen, während der Laufwerkhebel geöffnet war oder sich keine Diskette im Laufwerk befand.

[Diskette]

FD: Loading error
Während des Ladens von Daten trat ein Fehler auf.

FD: Not master
Die im Laufwerk 1 befindliche Diskette ist keine Masterdiskette.

Make ready FD:
Versuchter Zugriff auf Daten, während der Hebel nicht umgelegt war oder während sich keine Diskette im Laufwerk befand.

Anhang L BASIC-Fehlermeldungen

Wenn während des BASIC-Betriebs ein Fehler auftritt, erfolgt Anzeige der Fehlermeldung in einem der nachstehend aufgeführten Formate (1) oder (2).

(1) (Fehlertyp) error
(2) (Fehlertyp) error in (Zeilennummer)

Die Meldung (1) wird angezeigt, wenn ein Direktbefehl über die Tastatur eingegeben wird; die Meldung (2) wird während der Programmausführung angezeigt.

Fehler-Nr.	Angezeigte Meldung	Erklärung
1	Syntax error	Die eingegebene Anweisung entspricht nicht den Syntaxregeln von BASIC.
2	Overflow error	Die Größe eines numerischen Wertes überschreitet die gesetzten Grenzen.
3	Illegal data error	Es trat ein numerischer Wert oder eine Variable auf, der (die) nicht den numerischen Regeln von BASIC entspricht.
4	Type mismatch error	Daten- und Variablentyp stimmen nicht überein.
5	String length error	Die Anzahl der in einer Zeichenfolge enthaltenen Zeichen überschreitet 255.
6	Memory capacity error	Der zur Verarbeitung erforderliche Speicherraum reicht nicht aus.
7	Array def. error	Tritt auf beim Versuch, eine Feldvariable nachträglich zu vergrößern oder bei Auftreten eines nicht definierten Feldvariablennamens.
8	Line length error	Die Länge einer Zeile überschreitet die festgelegten Grenzen.
10	GOSUB nesting error	Die Anzahl der Verschachtelungen für eine GOSUB-Anweisung überschreitet 15.
11	FOR-NEXT error	Die Anzahl der Verschachtelungen für eine FOR-NEXT-Anweisung überschreitet 15.
12	DEF FN nesting error	Die Anzahl der Verschachtelungen für eine mit der DEF FN-Anweisung gegebene Funktionsdefinition überschreitet 6.
13	NEXT error	Auftreten einer NEXT-Anweisung ohne Vorliegen einer zugehörigen FOR-Anweisung.
14	RETURN error	Auftreten einer RETURN-Anweisung ohne Vorliegen einer GOSUB-Anweisung.
15	Un def. Function error	Eine nicht definierte Funktion wurde aufgerufen.
16	Un def. line error	Eine nicht vorhandene Zeilennummer oder Marke wurde spezifiziert.
17	Can't CONT error	Fortsetzung der Programmausführung mit der CONT-Anweisung ist unmöglich.
18	Memory protection error	Es wurde versucht, Daten in den Bereich für den BASIC-Interpreter zu schreiben.
19	Instruction error	Ein Direktbefehl ist im Programm enthalten.

Fehler-Nr.	Angezeigte Meldung	Erklärung
20	Can't RESUME error	Eine RESUME-Anweisung kann nicht verwendet werden.
21	RESUME error	Versuchte Ausführung einer RESUME-Anweisung, obwohl kein Fehler auftrat.
22	PAL error	Palettenblocknummer außerhalb des Bereichs.
24	READ error	Auftreten einer READ-Anweisung ohne entsprechende DATA-Anweisung.
29	Framing error	Zeichenbildungsfehler
30	Overrun error	Überlauffehler
31	Parity error	Paritätsfehler
40	File not found error	Versuch, auf eine nicht vorhandene Datei zuzugreifen.
42	Already exist error	Ein Versuch wurde unternommen einer Datei einen bereits vorhandenen Dateinamen zu geben.
43	Already open error	Versuch wurde unternommen, eine bereits geöffnete Datei zu öffnen.
44	Not open error	Versuchter Zugriff, Ausführung von CLOSE oder KILL für eine Datei, die nicht geöffnet wurde.
46	Write protect error	Ein Versuch wurde unternommen, Daten in eine schreibgeschützte Datei zu schreiben.
51	Too many files error	Es wurde versucht mehr als 32 Dateien auf der RAM-Speicherkarte zu speichern.
53	No file space error	Freie Bereich für Dateispeicherung sind nicht ausreichend.
58	DEV. name error	Ein ungültiger Gerätename wurde spezifiziert.
59	Can't execute error	Versuch, eine Anweisung für ein ungültiges Gerät auszuführen.
60	Illegal filename error	Ein ungültiger Dateiname wurde spezifiziert.
61	Illegal file mode error	Versuchter Zugriff auf eine Datei in einer nicht zulässigen Betriebsart.
63	Out of file error	Auftreten des Dateiendes während des Lesens von der Cassette.
64	Logical number error	Auftreten eines Fehlers in der Dateinummer.
65	LPT: not ready error	Kein Drucker angeschlossen oder dieser befindet sich nicht in der ON-LINE-Betriebsart. Auftreten von Betriebsstörungen im Druckermechanismus.
68	Dev. mode error	Auftreten eines Fehlers in der Geräte-Betriebsart.
69	Unprintable error	Auftreten eines Fehlers, für den keine Meldung vorgesehen ist.
70	Check sum error	Auftreten eines Fehlers in den Prüfsummendaten.

Anhang M Index

A
ABS 5-11
AND 5-9
ASC 5-14
ATN 5-12
AUTO 6-3
AXIS 6-92

B
BLINE 6-71
BOOT 6-47
BOX 6-71
BYE 6-47

C
CHAIN 6-56
CHR$ 5-14
CIRCLE 6-72
CLOSE # 6-63
CLR 6-7
CLS 6-10
COLOR 6-65
CONSOLE 6-10
CONT 6-8
COS 5-11
CSRH 5-6
CSRV 5-6
CURSOR 6-11

D
DEF FN 6-37
DEF KEY 6-39
DEFAULT 6-64
DELETE 6-4,55
DIM 6-33
DIR 6-48

E
END 6-14
EOF (#) 6-62
ERL 5-6
ERN 5-6
EXP 5-11

F
FOR ~ NEXT 6-15

G
GET 6-32
GOSUB ~ RETURN 6-20
GOTO 6-18
GPRINT 6-91

H
HCOPY 6-94
HSET 6-91

I
IF ~ THEN ~ :ELSE 6-22
IF ~ GOSUB 6-25
IF ~ GOTO 6-24
INIT 6-41
INP@ 6-97
INPUT 6-31
INPUT # 6-61
INT 5-11

K
KEY LIST 6-40
KILL # 6-63

L
LABEL 6-17
LEFT$ 5-12
LEN 5-14
LET 6-12
LIMIT 6-99
LINE 6-70
LIST 6-4
LIST/P 6-94
LN 5-12
LOAD 6-50
LOG 5-12

M
MERGE 6-57
MID$ 5-12
MUSIC 6-78

N
NEW 6-6
NEW ON 6-7
NOISE 6-83
NOT 5-9

O
ON ERROR GOTO 6-100
ON~GOSUB 6-21
ON~GOTO 6-19
OR 5-9
OUT@ 6-97

P
PAGE 6-87
PAI 5-12
PAINT 6-73
PAL 6-66
PATTERN 6-74
PCIRCLE 6-93
PCOLOR 6-86
PEEK 6-96
PHOME 6-90
PLINE 6-88
PLOT 6-95
PMODE 6-84
PMOVE 6-89
POINT 6-77
POKE 6-96
POSH 5-6
POSITION 6-75
POSV 5-6
PRINT 6-26
PRINT# 6-59
PRINT/P 6-87
PRINT/P USING 6-88
PRINT USING 6-27
PSKIP 6-86
PTEST 6-84

R
RAD 5-12
READ~DATA 6-34
REM 6-12
RENAME 6-55
RENUM 6-6
RESET 6-69
RESTORE 6-36
RESUME 6-101
RIGHT$ 5-12
RLINE 6-89
RMOVE 6-90
RND 5-14
ROPEN# 6-60
RUN 6-9,49

S
SAVE 6-52
SEARCH 6-5
SET 6-68
SIN 5-11
SIZE 5-6
SGN 5-11
SOUND 6-82
SPC 5-13
SQR 5-11
STICK 5-15
STOP 6-13
STR$ 5-13
STRIG 5-15
SYMBOL 6-76

T
TAB 5-13
TAN 5-11
TEMPO 6-82
TI$ 5-6
TROFF 6-39
TRON 6-38

U
USR 6-98

V
VAL 5-13
VERIFY 6-54

W
WAIT 6-47
WOPEN# 6-58

X
XOR 5-9

Anhang N Technische Daten

CPU	Z80A-CPU
Clock	3.5469 MHz
Speicher	ROM 16K byte RAM 64K byte VRAM 16K byte Kann auf 32K byte erweitert werden. (Zusatz)
Display	Schnittstelle : RF, Bildsignal, RGB Grafikanzeige : 320 × 200 Punkte 640 × 200 Punkte
Cassette	Standard-Audiocassettenband Datentransfergeschwindigkeit : 1200 Bit/sec. Datentransfersystem : SHARP PWM
Tasten-Layout	ASCII-Standard-Haupttastatur Spezialfunktionstasten Cursorsteuertasten Cassettenbanddecksteuertasten
Editierfunktion	Cursorsteuerung : nach oben, unten, links, rechts, Normalstellung, löschen Löschung, Einfügung
Clock-Funktion	Eingebaut
Stromversorgung	Ortsnetzspannung
Temperatur	Betriebstemperatur : 10° bis 35°C
Feuchtigkeit	20% ~ 80% (keine Feuchfigkeit)
Gewicht	MZ-811; 4.0 kg MZ-821; 4.3 kg
Abmessungen	Breite : 440 mm Tiefe : 305 mm Höhe : 109 mm
Zubehör	Netzkabel Monitorkabel Bediener-Handbuch Programmcassette Etiketten für die vom Benutzer definierbaren Tasten Graphiktasten-Etiketten

Printed in Japan
Gedruckt in Japan
Imprimé au Japon
Stampato in Giappone

© 1984 SHARP CORPORATION

4M 2 0-I(TINSG1212ACZZ)3

SHARP CORPORATION
OSAKA, JAPAN